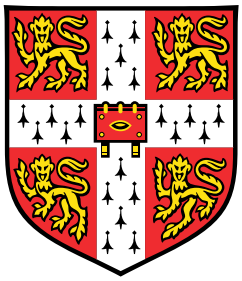
Machine Learning and Bayesian Statistics for Seismic Compressive Sensing



Georgios Pilikos

Department of Physics
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Darwin College July 2018

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. It is not substantially the same as any that I have submitted, or, is being concurrently submitted for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. It does not exceed the prescribed word limit for the relevant Degree Committee.

Georgios Pilikos
July 2018

Acknowledgements

This research was funded by the Engineering and Physical Sciences Research Council (EPSRC) and BP through an Industrial CASE studentship [1502944]. The synthetic data set is based on a model by SEAM-II and modelled by BP. The field data set is made available by New Zealand Petroleum and Minerals (NZPM). The research presented in this thesis appears partly or fully in journals and conference papers written with collaborators as stated in the Introduction. I would like to thank Anita Faul, Nikos Nikiforakis, Neil Philip, Raymond Abma and Mark Foster for their help during this work.

Abstract

Seismic surveys involve an artificial source of waves and a grid of receivers at the surface. Often, receivers could be missing either because they malfunctioned or could not be placed in certain locations. It could also be the fact that a local source of noise renders a receiver's output as unusable. These gaps in the data cause problems in later stages of the seismic signal processing work flow via aliasing or incoherent noise and thus signal reconstruction is necessary. Modern algorithms utilise the principle of Compressive Sensing (CS) for reconstruction which uses the assumption that the signal of interest is either sparse in nature or in some other bases. Most algorithms are designed with the only aim to fill in gaps in the data without any consideration of learning bases or quantifying uncertainty in their predictions.

In this thesis, we approach the seismic CS problem using probabilistic data-driven models that are adaptable to seismic data. We propose to use algorithms from the Bayesian statistics and machine learning field that allow the construction of models using probability distributions over random variables. This allows the modelling of sparsity and provides flexibility by adding or removing basis functions from the model. It also provides the framework for learning new dictionaries of bases, associating uncertainty for each prediction and denoising seismic signals. More specifically, we utilise two Bayesian algorithms for seismic CS, the Relevance Vector Machine (RVM) and the Beta Process Factor Analysis (BPFA).

The RVM uses a sparsity promoting distribution over the coefficients of a linear combination of basis functions. By learning the appropriate parameters, the algorithm infers a predictive mean and predictive variance that is used for prediction of receivers' values and uncertainty quantification. Experiments and comparisons on various seismic data show the effectiveness of the RVM with state-of-the-art reconstruction accuracy. Furthermore, its predictive variance is used along with modifications in order to create

uncertainty maps with varying levels of correlation between uncertainty and respective reconstruction error of receivers.

On the other hand, BPFA uses an alternative approach to enforce sparsity providing exact zero coefficients as opposed to the RVM. Another advantage is that it also learns the bases from the available data and provides denoising of seismic signals. Experiments and comparisons on seismic data show that the BPFA obtains state-of-the-art reconstruction accuracy on various domains. In addition, the learned bases are used by other algorithms to improve their performance. An analysis of the BPFA's inference procedure is given along with insights to reduce its computational cost. We also utilise the probabilistic nature of the BPFA and calculate the variance of the receivers' predictions obtained during inference. Using this, we create uncertainty maps that are highly correlated with the reconstruction error, obtaining better results than the RVM's predictive variance. Finally, an analysis of seismic signals with different levels of variance is undertaken in order to provide guidance for the best choice of algorithm per region.

The amount of seismic data available is growing, nevertheless quantity does not directly translate to quality. This creates the challenge to analyse and extract as much information and insight as possible. Using probabilistic data-driven models, we show how to achieve this by reconstructing seismic signals from under-sampled data, learn features from training data, denoise and create uncertainty maps for predictions in seismic surveys.

Table of contents

List of figures

List of tables

Nomenclature

Symbol	Description
K	Number of receivers in a patch
L	Number of features/bases
$\mathbf{x}^{(i)} \in \mathbb{R}^K$	Patch of receivers
$\mathbf{X} \in \mathbb{R}^{K \times T}$	Training data set
$\mathbf{k}^{(i)} \in \mathbb{R}^c$	Receiver's coordinates
$t_i \in \mathbb{R}$	Receiver's value
$\mathbf{w} \in \mathbb{R}^L$	Model's coefficients
$\mathbf{\Psi} \in \mathbb{R}^{M \times L}$	Fixed basis functions evaluated at entire domain
$\mathbf{\Phi} \in \mathbb{R}^{N \times L}$	Fixed basis functions evaluated at available data
$\mathbf{D} \in \mathbb{R}^{K \times L}$	Learned dictionary of bases
Q	Quality of reconstruction accuracy
$\mathcal{N}$	Normal Distribution

Beta	Beta Distribution
Gamma	Gamma Distribution
Bernoulli	Bernoulli Distribution

Abbreviation	**Description**
RVM	Relevance Vector Machine
BPFA	Beta Process Factor Analysis
SPGL1	Spectral Projected Gradient for L1
POCS	Projection Onto Convex Sets
CS	Compressive Sensing
DCT	Discrete Cosine Transform
FK	Frequency Wavenumber
SVD	Singular Value Decomposition

Chapter 1

Introduction

A seismic survey is an indispensable process for the geophysics community through which an image of the interior structure of the Earth is produced. An artificial source of body waves is used at the surface. This creates reflections from deep impedance changes at rock layer boundaries which are recorded by grids of receivers. In modern surveys, we have hundreds of source/receiver pairs that are used in the seismic processing work flow (Sheriff and Geldart, 1982) to produce three dimensional images of the subsurface. These images are used by academia and by the oil and gas industry for various purposes to study the Earth's structure for volcanic activity, for earthquakes, for plate tectonics and for the exploration of minerals, to name a few.

For the image of the subsurface to be useful, it has to be of sufficient resolution and quality. Nevertheless, in both land and marine seismic surveys, we frequently have receivers or groups of receivers missing either because receivers malfunctioned, or could not be placed in some locations. It could also be the fact that some local source of noise renders a receiver's output as unusable. Being able to obtain seismic signals from fewer receivers without significantly compromising their quality is not only essential when receivers are missing but has great importance for other reasons. Surveys which are better for the environment would be possible with the reduction of receivers. The potential minimisation of the duration of seismic surveys can also provide better health and safety conditions. In addition, financial gains are possible by lowering the seismic acquisition costs. The content of this thesis covers the problem of reconstructing seismic signals from fewer receivers. We will discuss the current state of this topic with its current limitations and propose a new approach towards its solution using probabilistic data-driven models.

1.1 Seismic Compressive Sensing and Acquisition

Compressive Sensing (CS) (Candes and Wakin, 2008; Donoho, 2006) is a framework that aims to reconstruct signals from a reduced amount of receivers without significantly compromising the signal's quality. It uses sparsity assumptions about the signals which means that the majority of the components are zero. The geophysics community has been using this framework (Mosher et al., 2012), more specifically *seismic CS*, to obtain seismic signals from fewer receivers by transforming the acquisition domain to a sparse domain. Dictionaries of basis functions such as the Fourier (Sacchi et al., 1998), the Radon (Trad et al., 2002), the curvelet (Herrmann and Hennenfent, 2008) and the focal (Kutscha and Verschuur, 2016) transform have been used in order to provide sparse representation for seismic CS. Nevertheless, by defining pre-fixed dictionaries, the algorithms are limited to represent the signal using the selected bases without taking into consideration that different instances of seismic signals vary and are not all successfully represented by one dictionary. Recently, another type of algorithm that learns the dictionary of bases from available data is used with applications to predicting missing values and denoising. An overview of seismic signal reconstruction algorithms can be found in section 2.5.

Under-sampling during a seismic survey is a bigger problem than just missing data since the receivers act as samplers and convert the seismic wave field from a continuous to a discrete form. The process of discretisation can result in aliasing if the sampling is inadequate (i.e. if the receivers' spacing is too sparse). *Aliasing* is a type of noise that corrupts the signal and makes it unusable in later stages of the seismic processing work flow (Naghizadeh and Sacchi, 2010). It can be detected when transforming the data into the Frequency Wavenumber (FK) domain (refer to section 2.2). This is one criterion that can be used to check whether the reconstruction accuracy of CS algorithms is sufficient.

The types of algorithms in seismic CS are mainly ad hoc and their only aim is to fill in the gaps or denoise signals. When making predictions about the receivers' values, it is also advantageous to provide a confidence or uncertainty about those predictions. An uncertainty map accompanying predictions helps quantify the risk associated with the acquired seismic data. In addition, it can help future seismic surveys when designing the placement of receivers emphasising on uncertain areas.

1.2 Machine Learning

In order to move away from ad hoc CS algorithms, it is desirable to better understand the generative process of the available data. One way to achieve this is to create data-driven

models that are adaptable when using training data. *Machine Learning* is a field that creates data-driven models using general modelling assumptions and adapts the model variables and parameters according to the data. There are different categories within machine learning: supervised learning, unsupervised learning and reinforcement learning. *Supervised learning* is the learning of models where the training data are labelled (i.e. input data with associated value). Typical problems in this category are: regression and classification. In this thesis, we will solve regression problems. *Regression* is the creation of a model from training data with the ability to predict values of unseen examples. Care is needed not to overfit the data (i.e. make the model work for the training data but not generalisable) or underfit by not providing enough examples. Once a model is learned, it can be used for various tasks such as prediction and denoising.

On the other hand, *unsupervised learning* is the learning of models that utilise raw training data and identify patterns/features to infer values for new data points. The training is done without any labels. Typical problems in this category are: clustering, density estimation and feature learning. In this thesis, we will solve feature learning problems. *Feature learning* is the learning of features/bases from raw data with algorithms adapting different configurations of the dictionary of bases to obtain the best fit for the given training data. In the context of Compressive Sensing (CS), the best choice is the dictionary of bases that provides sparse representation for the training data. Researchers have been using their domain expertise to design suitable basis functions for their specific application and careful engineering is necessary to identify those that model the data well. With feature learning, this is optimised for the given data by algorithms.

Lastly, *reinforcement learning* is the learning of models where agents take actions while maximising a reward (or a utility cost). This is a growing field in machine learning but we will not use it in this thesis. Interested readers refer to Sutton and Barto (1998) for further details.

1.3 Bayesian Statistics

The discussion so far has been given on general data-driven models. Nevertheless, if uncertainty information is desirable, probabilistic data-driven models can be constructed that are comprised of random variables. These variables incorporate assumptions about the data (i.e. sparsity in CS) and can be defined using various probability distributions. *Bayesian statistics* is a field that tackles this problem by using assumptions before observing the data and adapting the model's variables after obtaining the observations. Given data and assumptions about their generative process, a model is formulated and

then learned via an inference procedure and due to the random nature of the variables, it is possible to obtain predictions and uncertainties. In addition, it provides extra flexibility in modelling, since it is possible to incorporate prior knowledge. In this thesis, we propose to use two Bayesian models for seismic CS to create probabilistic data-driven models that are more adaptable to the data and also to utilise their uncertainty feature.

The first model that we will use is the *Relevance Vector Machine (RVM)* (Tipping, 2001; Tipping and Faul, 2003). This model uses a sparsity promoting prior distribution in the form of a hyper-prior over the coefficients of a linear combination of basis functions (refer to section 4.1 for further information). By learning the appropriate parameters, it can provide a predictive mean and predictive variance for the coefficients of the desired model. These can then be used for data prediction and uncertainty quantification. If continuous basis functions are used, predictions can be made everywhere.

The second model that we will use is the *Beta Process Factor Analysis (BPFA)* (Paisley and Carin, 2009; Zhou et al., 2012). This model uses a different approach to enforce sparsity in the coefficients of the linear combination of the desired model. This is achieved using a Bernoulli distribution to control whether a coefficient is zero or not. The parameter that controls the Bernoulli distribution is governed by a Beta distribution to allow flexibility in the level of sparsity. This is then element-wise multiplied with a normal distribution to produce the distribution of a desired coefficient (refer to section 5.1 for further details). This method of modelling provides exact zero coefficients as opposed to the RVM and allows extra flexibility by learning the bases from the available data. It infers the variables using Gibbs sampling which is an iterative procedure for Bayesian inference (refer to section 3.2). We utilise the probabilistic nature of the BPFA and calculate the variance of the predictions obtained from the Gibbs sampling process. Using this variance, it is possible to obtain uncertainty maps that are highly correlated with respective reconstruction errors.

1.4 Challenges

Moving away from ad hoc algorithms and using probabilistic data-driven models has its difficulties. The main challenges involved in seismic acquisition and concurrently in seismic Compressive Sensing (CS) are:

1. Reconstruction using fewer receivers is not trivial since under-sampling can cause aliasing. This distorts the signal and makes it unusable in the seismic processing work flow. The *reconstruction accuracy* has to be of sufficient resolution to avoid aliasing.

2. *Large gaps* often exist in seismic data acquisition due to various environmental, physical or financial limitations. Interpolating when consecutive receivers are missing is a challenge since information from neighbouring receivers is constrained.

3. Sometimes receivers malfunction or are contaminated by different types of noise such as interference from other sources or surveys. It could also be the fact that other environmental signals are picked up. *Removing this noise* and obtaining the accurate seismic signal is not trivial.

4. The majority of the techniques use *predefined dictionaries of basis functions* to describe the underlying signals. However, this is very limiting since different signals could contain different structures that require their own sparse representation. Thus, there is a challenge to find the appropriate dictionary of basis functions from all possibilities for the signal at hand.

5. Not only is there a choice about which basis functions to use to describe the underlying seismic wave field, but also there is a plethora of possibilities for the functionality of the algorithms themselves. Each algorithm has its own *parameter settings that need to be tuned* for a particular application or a particular seismic signal. Some examples include the patch size that they operate, the number of iterations to run and the size of the dictionary of bases.

6. Different choices result in different reconstruction accuracies and different computational times. *Fast processing times* are important since CS algorithms operate on large volumes of seismic data and usually in many dimensions. The speed of operation is useful not only for faster results but also to allow for fast experimentation and tuning of parameters.

7. Reconstructing the seismic wave field involves the prediction of unknown values for receivers. However, it could be that some predictions are more accurate than others and the algorithms should be able to provide a confidence level associated with each prediction. That is, an *uncertainty value* for each prediction is advantageous in order to provide the risk associated with a reconstruction.

8. Seismic signals vary with respect to their variance and structure. Different regions of a signal can have different variance and an algorithm's performance can vary. *Knowing when to use which algorithm under certain circumstances and criteria* is challenging.

1.5 Research Contributions

In this thesis, we address the above challenges in seismic CS using probabilistic data-driven models from the Bayesian statistics and machine learning literature. Our research contributions are the following:

1. We propose the application of the Relevance Vector Machine (RVM) for seismic CS. Using the RVM, we create a probabilistic data-driven model using a dictionary of basis functions. With appropriate choice of bases and parameters, we obtain state-of-the-art reconstruction accuracy with no signs of aliasing or noise in the Frequency Wavenumber (FK) domain.

2. We propose the utilisation of Beta Process Factor Analysis (BPFA) for seismic CS. With BPFA, we learn various dictionaries of bases that we can then use with other algorithms. We obtain state-of-the-art reconstruction accuracy with no signs of aliasing or noise. We evaluate the BPFA for the reconstruction of irregular and missing data with large gaps such as blocks and artificial rivers.

3. We use BPFA not only to predict missing receivers' values but also to denoise seismic signals. The BPFA creates a probabilistic data-driven model that is able to identify the presence of noise which stops the reconstruction earlier in order to ignore the noise in the model.

4. We propose the utilisation of the Gibbs sampler's variance during the BPFA inference in order to obtain uncertainty maps for seismic CS. We compare these with others in the literature and show that we can produce highly correlated uncertainty and error using the Spearman's correlation coefficient. This illustrates that it is possible to capture the reconstruction error and help in the design of seismic surveys.

5. We propose a Gibbs sampling analysis and investigate the effect that initialisation has on the inference procedure of BPFA both for reconstruction accuracy and uncertainty quantification. Using this, we are able to speed up BPFA reducing the amount of computation necessary.

6. We propose to use two hybrid algorithms in seismic CS that first learn a dictionary of bases using BPFA and then the RVM and the Spectral Projected Gradient for L1 (SPGL1) are used with these bases. By using these hybrids, it is possible to obtain high reconstruction accuracy with no aliasing or noise and at the same time provide fast computational time.

7. We propose a signal variance analysis using the Spearman's correlation coefficient. We calculate the variance of available data and monitor how the reconstruction accuracy changes with varying signal characteristics. Using this, we can get insight as to when an algorithm performs better and when another algorithm should be used obtaining better reconstruction accuracy overall.

8. We provide a comprehensive parameter study for various algorithms in seismic CS that can help future endeavours of tuning parameters. Using these, we provide detailed comparisons on thousands of seismic signals that provide further insight to each configuration.

9. During all experiments, we propose to perform seismic CS per time sample of the acquired signal in a receiver grid. By using this time slice processing approach, we show that we obtain better reconstruction accuracy due to the fact that the gaps in the data are smaller. To visualise any potential aliasing or noise, we re-sort the data into the x-t domain and then to the FK domain.

The research presented in this thesis appears in the following journal and conference papers, partly or fully and machine learning theory appears in lecture notes written by the author while lecturing for the MPhil in Scientific Computing at the Department of Physics of the University of Cambridge:

Journals

- "The Relevance Vector Machine for seismic Bayesian compressive sensing", **G. Pilikos**, *under review.*

- "Bayesian modelling for uncertainty quantification in seismic compressive sensing", **G. Pilikos** and A. C. Faul, *GEOPHYSICS, 84(2), P15-P25, 2019*, doi:10.1190/geo2018-0145.1 .

- "Bayesian feature learning for seismic compressive sensing and denoising", **G. Pilikos** and A. C. Faul, *GEOPHYSICS, 82(6), O91-O104, 2017*, doi:10.1190/geo2016-0373.1.

Conferences

- "Beta Process Factor Analysis for Efficient Seismic Compressive Sensing with Uncertainty Quantification", **G. Pilikos** and N. Philip, *IEEE International Conference on Digital Signal Processing (DSP), 2018*, doi:10.1109/ICDSP.2018.8631841.

- "Seismic compressive sensing beyond aliasing using Bayesian feature learning", **G. Pilikos**, A. C. Faul and N. Philip, *SEG Technical Program Expanded Abstracts: pp. 4328-4332, 2017*, doi:10.1190/segam2017-17558742.1.
- "Relevance Vector Machines with Uncertainty Measure for Seismic Bayesian Compressive Sensing and Survey Design", **G. Pilikos** and A. C. Faul, *IEEE International Conference on Machine Learning and Applications (ICMLA), 2016*, doi:10.1109/ICMLA.2016.0166.
- "The model is simple, until proven otherwise - how to cope in an ever changing world", A. C. Faul and **G. Pilikos**, *Data for Policy, Frontiers of Data Science for Government, 2016*, doi:10.5281/zenodo.556502.

Lecture Notes

- "Unsupervised Learning", **G. Pilikos**, *Lecture notes for the Machine Learning course*, part of the MPhil in Scientific Computing, Department of Physics, University of Cambridge, 2015-2017.

1.6 Thesis outline

The remainder of the thesis is structured as follows: In chapter 2, an overview of seismic data and the domains used is given along with descriptions of the data sets that will be used. We will also give an introduction to the aliasing problem along with illustrations of aliased seismic data and incoherent noise. An introduction to Compressive Sensing (CS) and various dictionaries of basis functions will be given next and then more specifically an overview of seismic CS algorithms.

In chapter 3, we will introduce the field of machine learning and Bayesian statistics. The framework of Bayesian modelling is described along with inference algorithms that we will use in this thesis. A description of the probability distributions that we will use is given along with descriptions of latent variable models. Finally, a detailed description of Bayesian regression is provided.

Then, in chapter 4 we will introduce the RVM for seismic CS and describe its fast version used throughout the thesis. Extensions of the RVM are discussed along with parameter tuning for various algorithms. A comparison of the best configuration of algorithms is given with respect to reconstruction accuracy and computational time. Finally, examples of reconstructions are provided using various algorithms and configurations on synthetic and field data.

In chapter 5, we will introduce the BPFA model for seismic CS. We provide a detailed description of its inner workings along with its patch processing procedure. Parameter settings are discussed along with initialisation of variables. We then provide examples of reconstructions along with learned bases. Comprehensive comparisons with other algorithms are provided illustrating the importance of learning dictionaries of bases. We then discuss how we can improve other algorithms using the learned bases. We show results for two hybrid algorithms: the SPGL1 and the RVM methods using BPFA bases and show the gains in reconstruction accuracy and computational time. The computational complexity and running times are also discussed. We then describe a Gibbs analysis of the BPFA inference procedure. Using this, we illustrate the potential speed up. Further tests with missing rivers and blocks are provided. An example of 3D interpolation is also illustrated. Finally, we show how the BPFA model can be used for denoising and we compare it with another algorithm in the literature.

In chapter 6, we will evaluate the reconstructions from the BPFA and the RVM in the x-t domain. We will re-sort the time slice reconstructions in this domain and then perform Frequency Wavenumber (FK) analysis for signs of aliasing or noise in reconstructions. Comparisons with other algorithms are given illustrating the performance of our proposed methods. Discussion for the reconstruction accuracy of seismic signals is given obtaining different accuracy in different regions. A variance analysis is also provided, showing the reconstruction accuracy of algorithms using training data with different variance.

In chapter 7, we will discuss how we can use the probabilistic data-driven models proposed for uncertainty quantification in seismic CS. We will use both the BPFA's Gibbs variance and the RVM's predictive variance. We also discuss modifications of the RVM and how we can use them to create uncertainty maps. We then show representative uncertainty maps and compare them using the Spearman's correlation coefficient (correlation between reconstruction error and uncertainty). In addition, we provide a variance analysis and show how this coefficient changes with different levels of variance in the available data. Furthermore, stacking of uncertainty maps is included showing the improvements gained from averaging signals.

Finally in chapter 8, we provide the conclusions of our findings and how future work can enhance this research area.

Chapter 2

Seismic Compressive Sensing and Acquisition

Seismic data acquisition involves sampling the seismic wave field at or near the Earth's surface. A source at the surface creates a wave field that is reflected and refracted by changes in impedance. Surface receivers record the reflected wave field generally on a regular grid. But some of those receivers may be missing, caused either by malfunction or because they could not be placed in the required location. Signal reconstruction algorithms are used in order to replace or restore the output of the missing receivers. Most of the modern algorithms use the principle of Compressive Sensing (CS) which uses the assumption that the signal of interest is either sparse in nature or in some other basis. In this chapter, a description of seismic data is given along with a description of the aliasing problem. We will then give an overview of the field of CS for seismic acquisition.

2.1 Seismic data

We will use two data sets in this thesis. The first data set and the one that most of the experiments are undertaken is synthetic and called SEAM-II (SEG, 2018a), provided by BP. It contains an artificial source (shot) and a grid of receivers. Thus, we will work in the *common-shot* domain where a gather contains the output of all receivers obtained from the same source. On the other hand, a *common-receiver* is when the gather contains the output of a particular receiver from multiple sources. All algorithms that we present in this thesis are applicable to both types of gathers but we will focus the experiments on the common-shot domain. The second data set that we will use is a field data set called Parihaka. It is a 3D seismic image provided for use by New Zealand Petroleum and Minerals (NZPM) and obtained from the SEG wiki (SEG, 2018b).

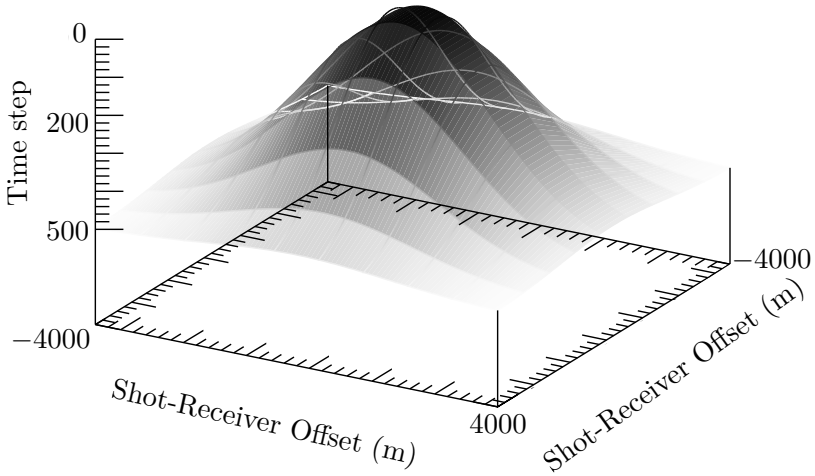


Fig. 2.1 This is a plot of a multivariate normal distribution to help the discussion. The shot-receiver offset axes correspond to the receivers' spatial coordinates and the other axis to time. For the x-t domain, we keep only one of the shot-receiver coordinates constant, giving a 2D signal with time on one axis and respective coordinate on the other. In a time slice, both receiver coordinates are used and time is kept constant.

There are different domains for processing seismic data, two of which are: the x-t domain and the time slice (x-y domain). In order to understand what each means, an example of 3D data is illustrated in Figure 2.1. The three axes illustrate the spatial coordinates of the receivers and time. Figure 2.2 shows a configuration where a central source is surrounded by receivers. Each grid node is a receiver, recording the reflected wave field generated by the source. The *x-t domain* is comprised of a line of receivers acting during a single source release. Figure 2.3 shows a subset from a fixed coordinate. The time series output of each receiver is called a *trace*. This concept is illustrated in Figure 2.4 where a section from the x-t domain is magnified and individual traces can be seen. If we zoom-in to an individual trace, then we can see the variations more clearly as can be seen in Figure 2.5. A *time slice*, on the other hand, is when time is kept constant and the signal is viewed over all receivers at a particular time instant as seen in Figure 2.6. All these examples were extracted from the SEAM-II data set. The data set has a 6.25 metres spatial sampling. There are 1281 receivers along each line and there are 1281 lines spanning 8000 metres covering vertical and horizontal directions. The temporal sampling is 0.006 seconds and each receiver's trace has 500 time samples resulting in 3 seconds of recordings. Using this dataset, we will extract various time slices and x-t domains to evaluate different algorithms. The Parihaka data set will only be used for verification that the algorithms can operate on field data.

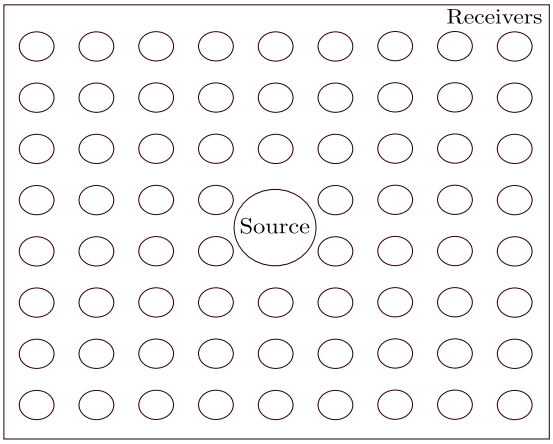


Fig. 2.2 Receiver grid with an artificial source in the middle of the domain.

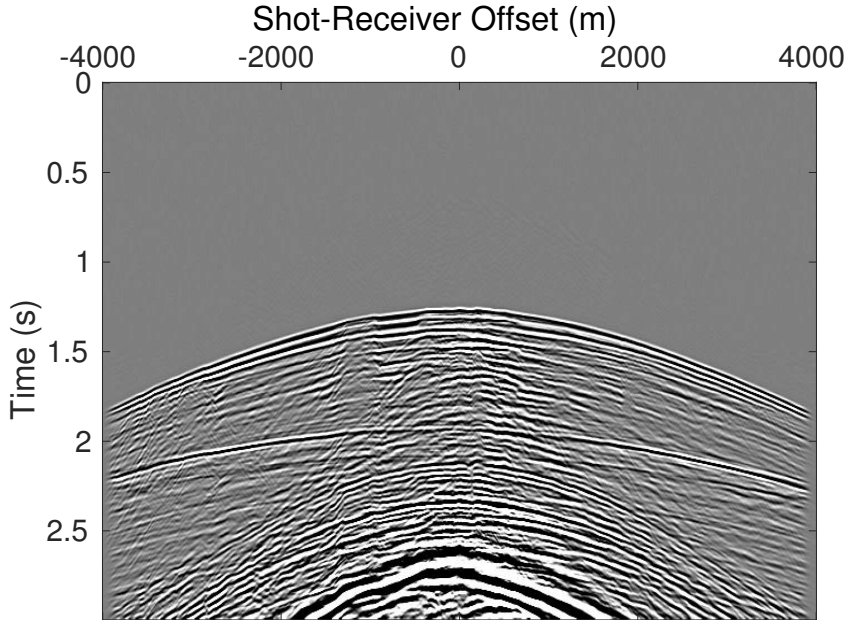


Fig. 2.3 x-t domain of seismic data from the SEAM-II synthetic data set.

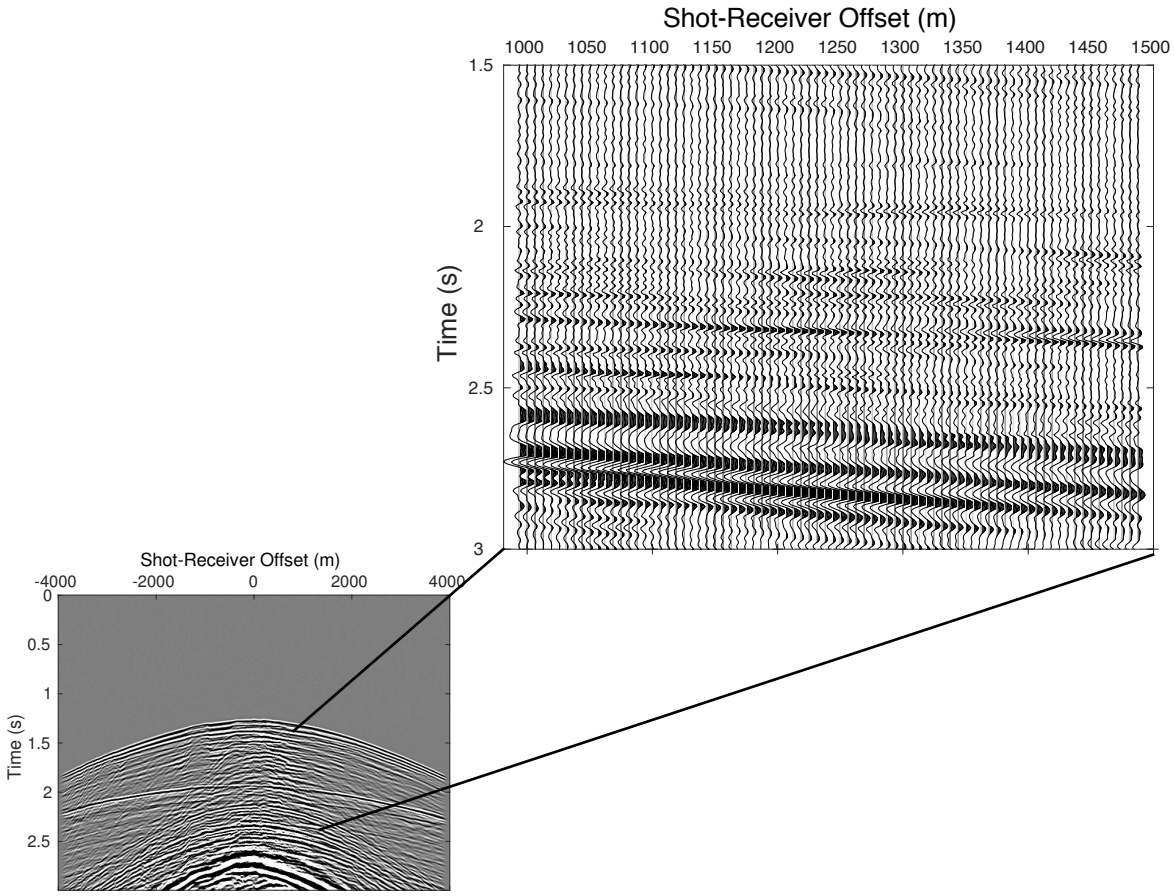


Fig. 2.4 A collection of multiple traces in x-t domain.

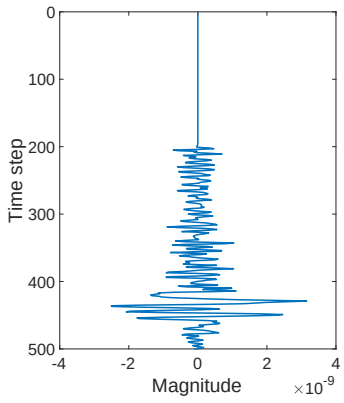


Fig. 2.5 A signal from a single trace showing the output of a receiver.

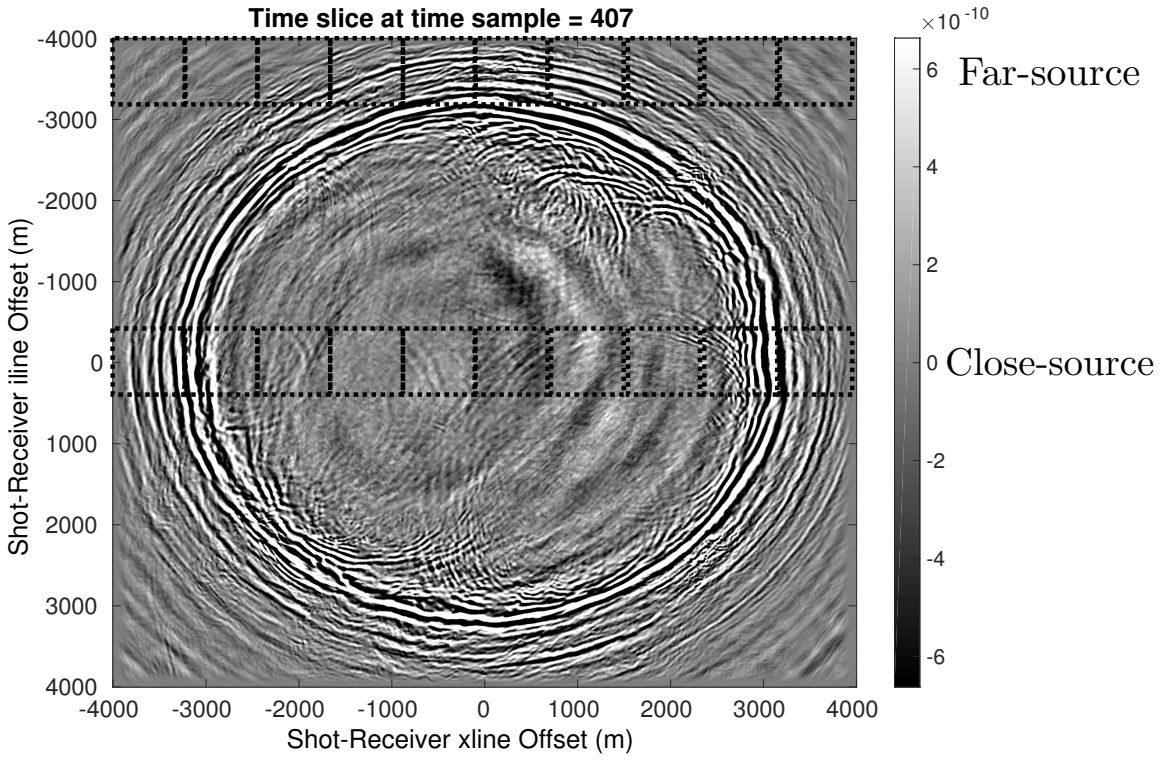


Fig. 2.6 An example of a time slice. It also includes an illustration of the location of close to source and far from source receiver lines in a time slice.

2.1.1 Close to and far from the source

As we discussed, the x-t domain is a line of receivers and depending on the location of the receivers in the grid, it can have different signal structure. In this thesis, we will examine x-t domains that contain receivers located close to and far from the source. We will be working with sections of time slices which when combined together, create entire receiver lines (sections 4.4.1 and 6.1 explain why we chose to operate on time slices). An illustration of the locations can be seen in Figure 2.6 with close to and far from the source receiver lines indicated. A receiver line far from the source was illustrated in Figure 2.3 and one close to the source can be seen in Figure 2.11(a). We can see that far from the source, the seismic signal starts much later and is smoother as opposed to the x-t domain closer to the source. The latter has a much steeper structure. We will examine how different algorithms behave under different circumstances in chapter 6.

2.2 Sampling and aliasing

During a seismic survey, the receivers act as samplers and record the seismic wavefield from a continuous to a discrete form. The process of discretisation can result in aliasing if the sampling is inadequate (i.e. if the receivers' spacing is too sparse) which makes the signal unusable in later stages of the seismic processing work flow (Naghizadeh and Sacchi, 2010). A domain that effectively detects aliased data is the Frequency Wavenumber (FK) domain. This is obtained using the Fourier transform of the x-t domain and we will use it in this thesis to partly evaluate reconstructions from various algorithms. In order to facilitate later discussions, an introduction to the FK domain will be given here along with examples of aliasing. The definitions and derivations are adaptations from here[1] by Morrison (2013).

Consider a continuous signal, i.e. a seismic wave field $w(t)$, with its Fourier transform given by

$$W(\omega) = \int_{-\infty}^{+\infty} w(t) e^{-j\omega t} dt, \tag{2.1}$$

and its inverse Fourier transform given by

$$w(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} W(\omega) e^{j\omega t} d\omega. \tag{2.2}$$

The act of sampling can be described by the product of the continuous signal with a periodic train of impulses that occur at samples every Δt period. That is, consider the function,

$$S(t) = \sum_{n=-\infty}^{+\infty} \delta(t - n\Delta t), \tag{2.3}$$

where $\delta(t)$ is the Dirac (or delta) function which is equal to zero everywhere except at $t = 0$ which is infinity. The product of the two functions is given by,

$$o(t) = S(t)w(t) \tag{2.4}$$

$$= \sum_{n=-\infty}^{+\infty} w(t)\delta(t - n\Delta t) \tag{2.5}$$

$$= \sum_{n=-\infty}^{+\infty} w(n\Delta t)\delta(t - n\Delta t). \tag{2.6}$$

[1] http://digitalcommons.mtech.edu/elec_engr_book/2 accessed 12 March 2018

The Fourier transform of the product is given by,

$$O(\omega) = \int_{-\infty}^{+\infty} o(t)e^{-j\omega t}dt \tag{2.7}$$

$$= \int_{-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} w(n\Delta t)\delta(t - n\Delta t)e^{-j\omega t}dt \tag{2.8}$$

$$= \sum_{n=-\infty}^{+\infty} \int_{-\infty}^{+\infty} w(n\Delta t)\delta(t - n\Delta t)e^{-j\omega t}dt \tag{2.9}$$

By considering the fact that the Dirac function is zero everywhere except $n\Delta t$ and that its integral over $-\infty$ and $+\infty$ results to 1,

$$O(\omega) = \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-j\omega n\Delta t}. \tag{2.10}$$

If we multiply by Δt,

$$Y(\omega) = \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-j\omega n\Delta t}\Delta t. \tag{2.11}$$

This expression is very similar to the continuous Fourier transform in 2.1 as $\Delta t \to 0$. Let us consider $\omega = 0$,

$$W(0) = \int_{-\infty}^{+\infty} w(t)dt \tag{2.12}$$

and

$$Y(0) = \sum_{n=-\infty}^{+\infty} w(n\Delta t)\Delta t. \tag{2.13}$$

As Δt gets smaller and smaller, $W(0) = Y(0)$. In the case of a different frequency, $\omega = \frac{2\pi}{\Delta t}$,

$$W(\frac{2\pi}{\Delta t}) = \int_{-\infty}^{+\infty} w(t)e^{\frac{-j2\pi}{\Delta t}t}dt \tag{2.14}$$

and

$$Y(\frac{2\pi}{\Delta t}) = \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-j\frac{2\pi}{\Delta t}n\Delta t}\Delta t \tag{2.15}$$

$$= \sum_{n=-\infty}^{+\infty} w(n\Delta t)\Delta t \tag{2.16}$$

$$= Y(0). \tag{2.17}$$

This means that the discrete Fourier transform of the product appears periodic. For an arbitrary frequency, $\Delta\omega$,

$$Y(\Delta\omega) = \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-j\Delta\omega n\Delta t}\Delta t \tag{2.18}$$

and

$$\begin{aligned} Y(\Delta\omega + \frac{2\pi}{\Delta t}) &= \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-j(\frac{2\pi}{\Delta t}+\Delta\omega)n\Delta t}\Delta t && (2.19)\\ &= \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-jn2\pi}e^{-j(\Delta\omega)n\Delta t}\Delta t && (2.20)\\ &= \sum_{n=-\infty}^{+\infty} w(n\Delta t)e^{-j(\Delta\omega)n\Delta t}\Delta t && (2.21)\\ &= Y(\Delta\omega). && (2.22) \end{aligned}$$

This periodicity can be explained if we consider the fact that multiplication in the time domain as in equation 2.4 can be regarded as a convolution in the frequency domain,

$$Y(\omega) = W(\omega) * S(\omega). \tag{2.23}$$

In the case of $s(t)$ as a train of impulses, the corresponding Fourier Transform, $S(\omega)$ is also given by a periodic train of impulses in the frequency domain. Therefore, a convolution of a periodic train of impulses with $W(\omega)$ results in a periodic frequency spectrum.

2.2.1 Aliasing

Aliasing occurs when this periodic frequency spectrum is not well spaced and the spectra overlap with each other. The spacing depends on the sampling rate (i.e. the frequency of the train of impulses from $S(t)$) and to avoid aliasing, the sampling rate should be greater than twice the highest frequency in the signal, called the *Nyquist rate*. An illustration of this can be seen in Figures 2.7 - 2.10 where ω_n is the largest frequency present in the band limited signal and ω_s is the sampling frequency. If $\omega_s \geq 2\omega_n$ then there is no aliasing as it can be seen in Figure 2.9. On the other hand, if $\omega_s < 2\omega_n$ then aliasing occurs as seen in Figure 2.10.

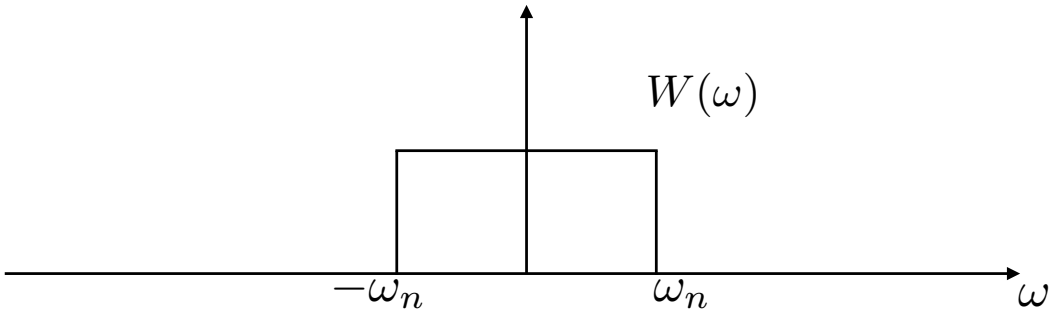


Fig. 2.7 An illustration of the Fourier Transform of a signal.

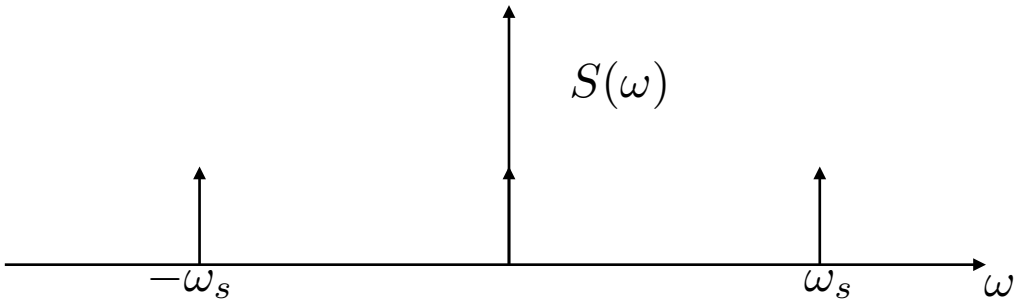


Fig. 2.8 An illustration of the Fourier Transform of a train of impulses.

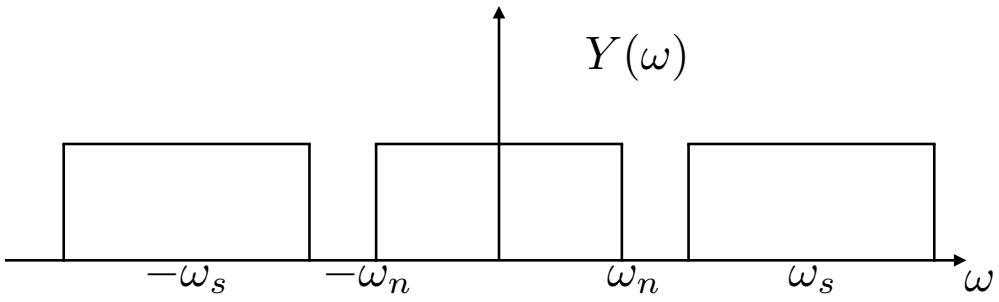


Fig. 2.9 No aliasing occurs with appropriate sampling rate.

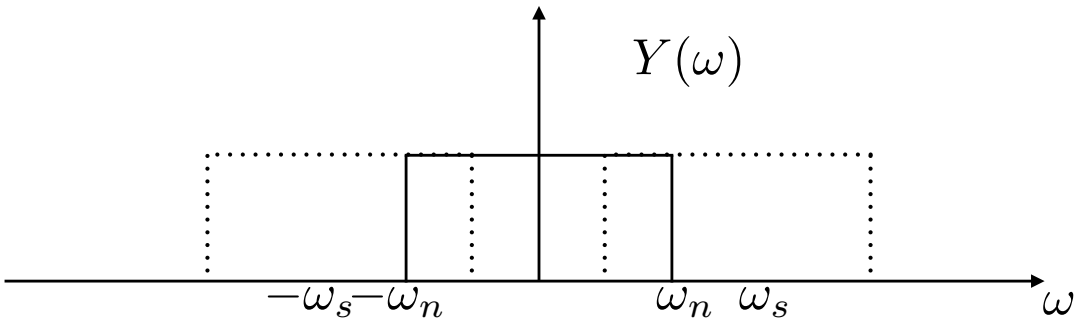
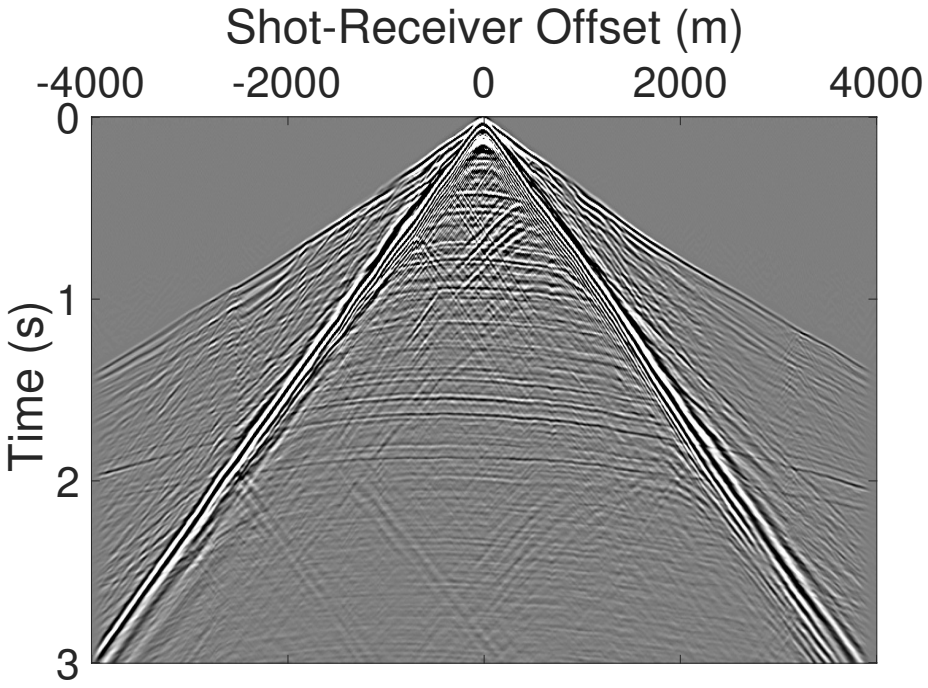


Fig. 2.10 Aliasing occurs with inadequate sampling rate.

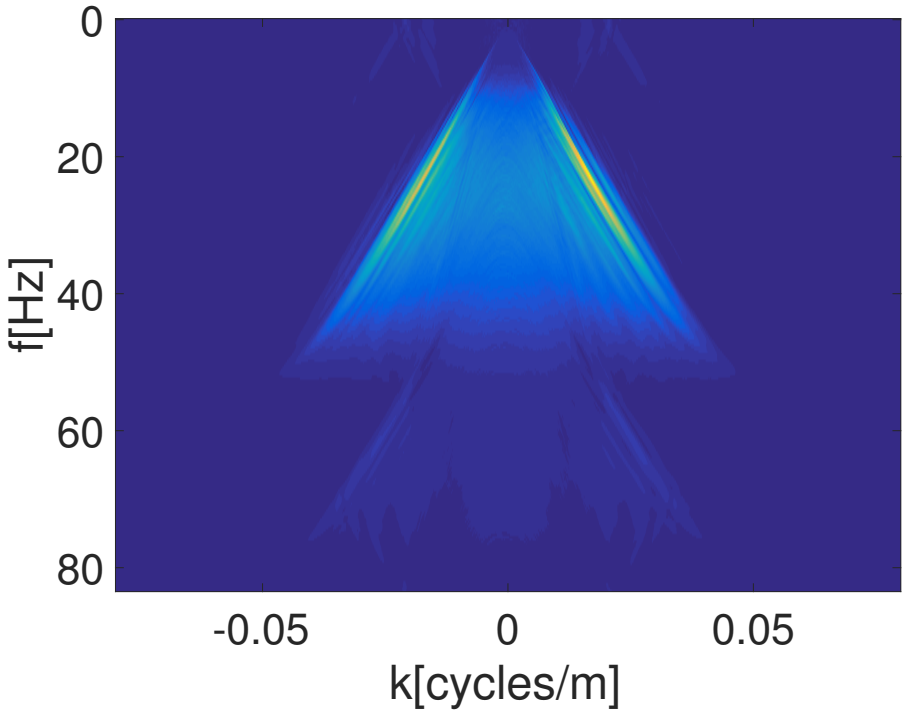
If we do not know a priori, the maximum frequency of the signal, the *Nyquist frequency* can be used to assess whether a signal would be aliased. This frequency is equal to half of the sampling rate and if there are frequencies in the signal higher than this, we would expect aliasing.

We will illustrate the aliasing effect in the seismic x-t domain extracted from the SEAM-II data set described earlier. Figure 2.11(a) shows the x-t domain that is spatially sampled at 6.25 metres per receiver. The Nyquist frequency is equal to 0.08 as illustrated in the axis of the corresponding Frequency Wavenumber (FK) domain seen in Figure 2.11(b). The FK domain illustrates the Fourier transform in time and in spatial direction. This shows that the Fourier spectra do not overlap with each other which means that the sampling rate of 6.25 metres is sufficient. That is, there are no frequencies higher than the Nyquist frequency. Figure 2.12(a) shows the same signal but this time sampled at 12.5 metres per trace. By changing the sampling rate, we can see that the Nyquist frequency changes to 0.04 in Figure 2.12(b).

It can be seen that the Fourier spectrum is now more spread out but not yet aliased as there is no overlap from the next spectrum. We continue the illustration in Figure 2.13(a) where the same signal is spatially sampled at 25 metres per receiver. Its corresponding FK domain can be seen in Figure 2.13(b). This illustrates that the Fourier spectrum is aliased from the adjacent Fourier spectra and thus it is overall aliased. The sampling rate is not sufficient and thus we would need to increase the sampling resolution. This is done by signal reconstruction algorithms that predict missing receivers. Nevertheless, there are other ways that we could sample the seismic wave field that can help avoid aliasing and we describe one next.

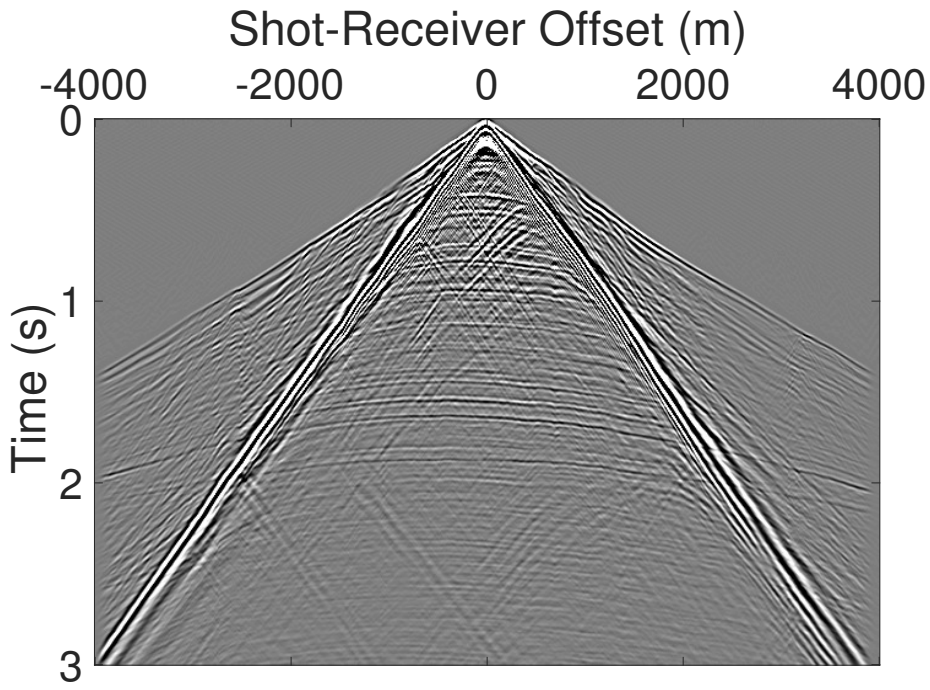


(a) Original, 6.25m sampling

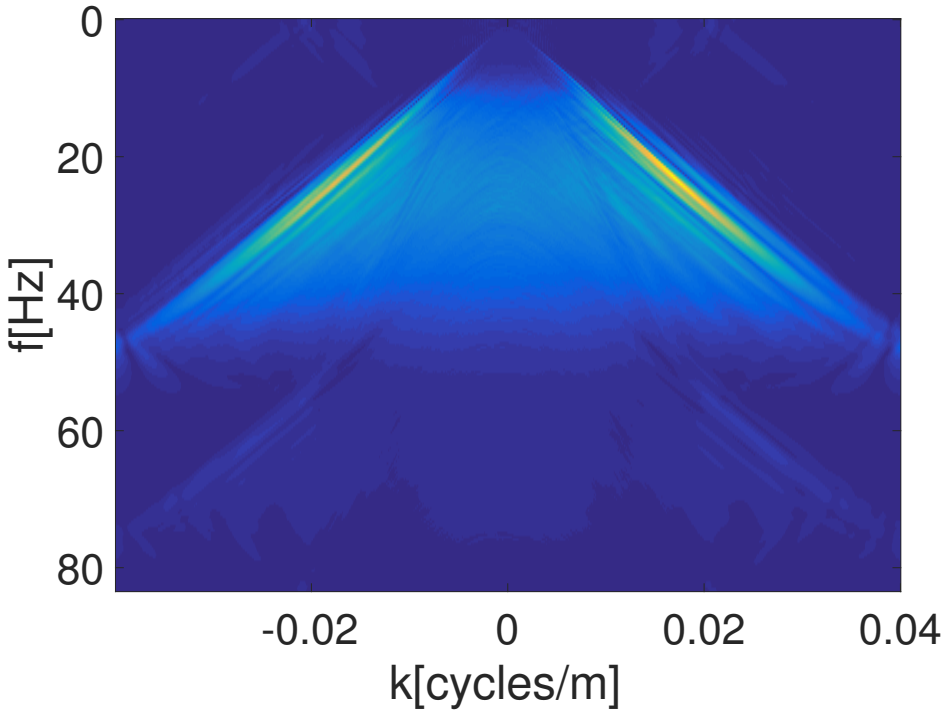


(b) FK, 6.25m sampling

Fig. 2.11 An example of x-t domain from the SEAM-II with its FK domain. (a) shows the original sampled at 6.25 metres per trace. The corresponding FK domain can be seen in (b).

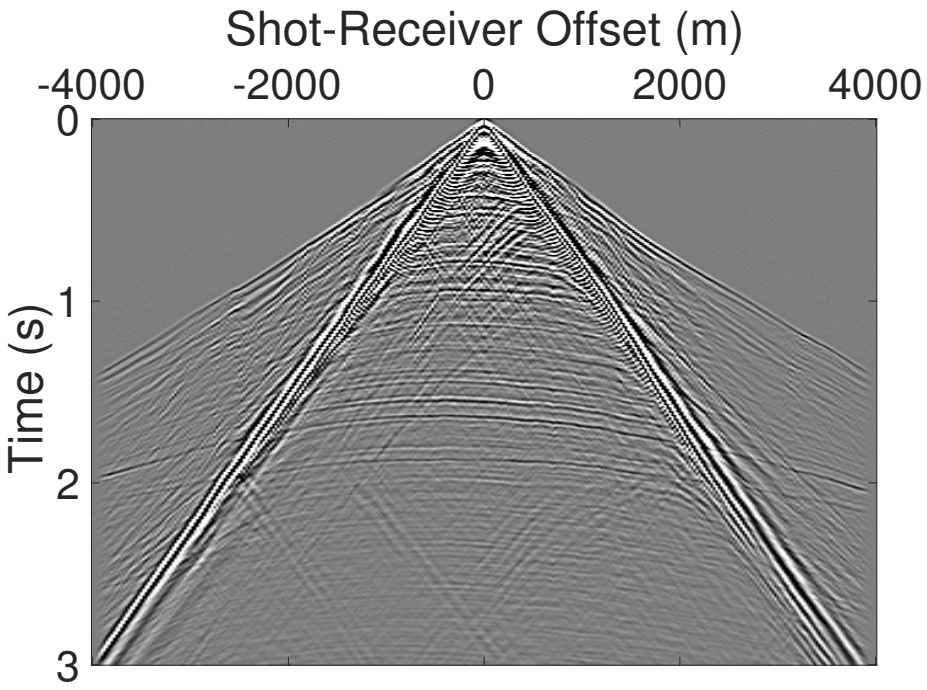


(a) 12.5m sampling

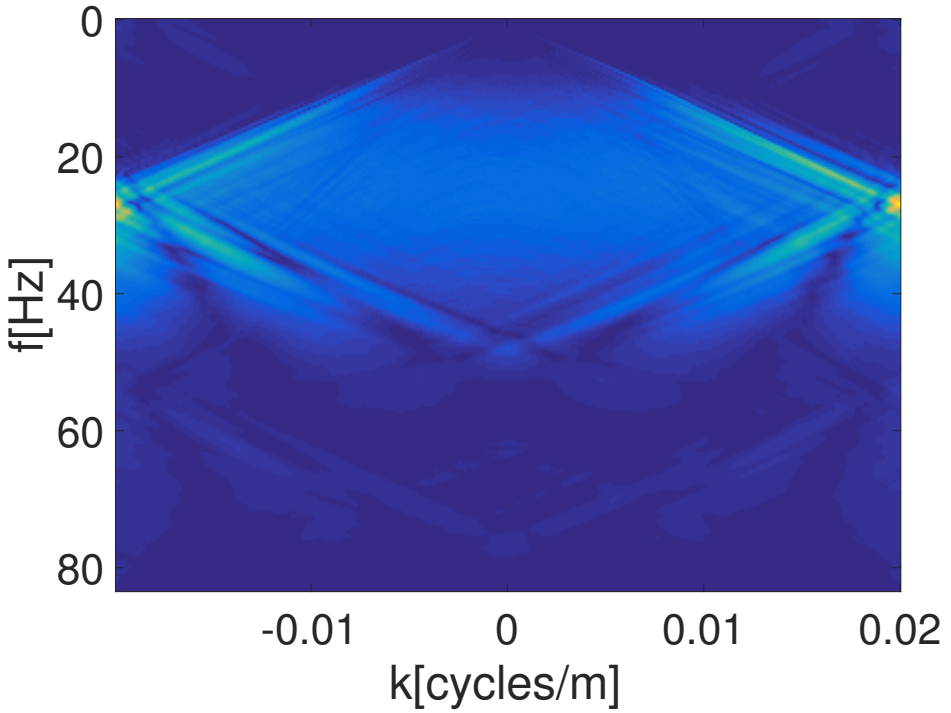


(b) FK for 12.5m sampling

Fig. 2.12 An example of x-t domain from the SEAM-II with its FK domain. (a) shows the signal of Figure 2.11(a) sampled at 12.5 metres per trace. The corresponding FK domain can be seen in (b).

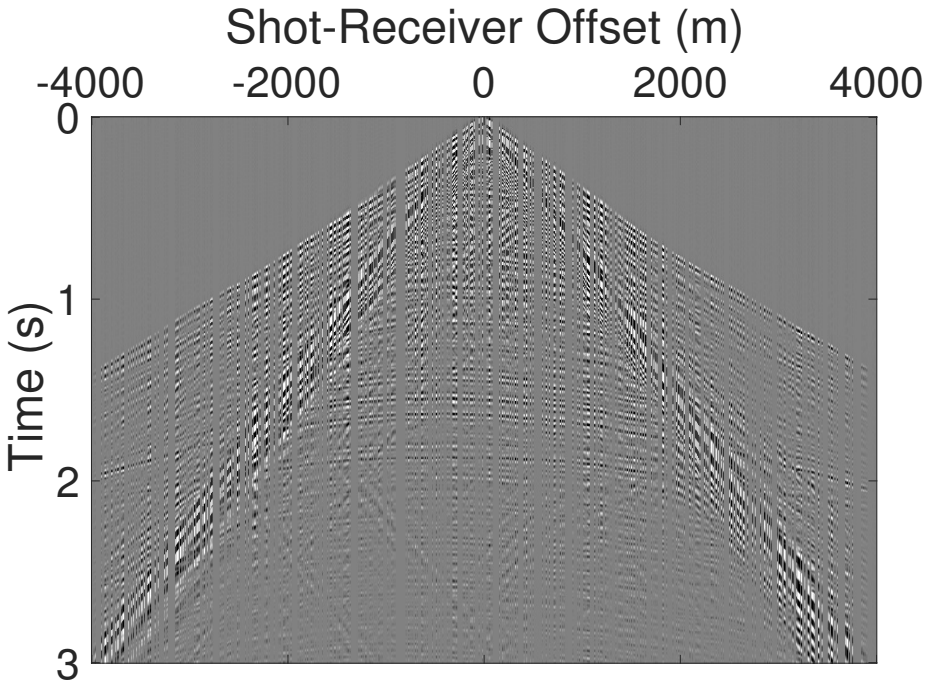


(a) 25m sampling

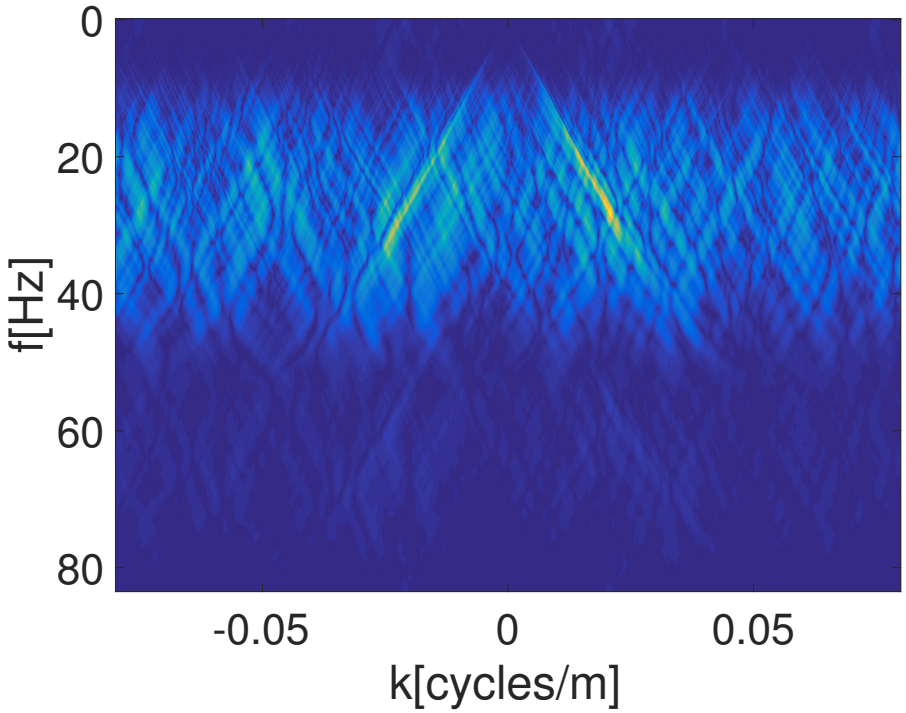


(b) FK for 25m sampling

Fig. 2.13 An example of x-t domain from the SEAM-II with its FK domain. (a) shows the signal of Figure 2.11(a) sampled at 25 metres per trace. The corresponding FK domain can be seen in (b).



(a) Using 30% of receivers



(b) FK from only 30% of receivers

Fig. 2.14 An example of x-t domain from the SEAM-II with its FK domain. (a) shows the signal of Figure 2.11(a) using only 30% of receivers randomly. The corresponding FK domain can be seen in (b).

2.2.2 Irregular sampling to avoid aliasing

An alternative to regular spacing of receivers is irregular sampling. That is, instead of placing the receivers in a regular grid, we can place the receivers at random or carefully selected locations spread around the domain. Figure 2.14(a) shows an example of a seismic signal sampled irregularly with 30% of the receivers used. The number of receivers is approximately equivalent to the receivers in the 25 metre regular sampling. Note that missing receivers are replaced with zeros. It can be seen that the FK domain in Figure 2.14(b) does not have other strong Fourier spectra but rather noise is introduced. This was also shown by Kumar et al. (2015) where periodic subsampling created aliasing as opposed to random subsampling that turned the aliases into incoherent noise. We will use random irregular sampling throughout the thesis to distort the signals of interest and aim to reconstruct unaliased equivalents. Using this type of sampling is both advantageous for the minimisation of aliasing but at the same time we will show that it is advantageous for the algorithms that we will be using due to the pattern of removal of the receivers.

2.3 Introduction to Compressive Sensing

We have seen that regularly under sampled signals can cause aliasing and randomly under sampled signals avoid it but introduce incoherent noise (Kumar et al., 2015) in the FK domain. In order to avoid noise, we need to predict (or interpolate) the missing receivers' values. To achieve it, we will use the Compressive Sensing (CS) framework (Candes and Wakin, 2008; Donoho, 2006).

CS allows signal reconstruction using N receivers with $N \ll M$ where M are all the receivers in the original signal of interest. These receivers are described by

$$\mathbf{t} = \mathbf{\Omega x}, \tag{2.24}$$

where $\mathbf{x} \in \mathbb{R}^M$ is the original collection of receivers, $\mathbf{t} \in \mathbb{R}^N$ is known as the collapsed signal that contains the reduced number of receivers and $\mathbf{\Omega} \in \mathbb{R}^{N \times M}$ is the sensing matrix that indicates where we have sensed. A necessary assumption is that the signal of interest is either sparse in nature or in some basis. Let $\mathbf{w} \in \mathbb{R}^L$ be the sparse signal and defined by

$$\mathbf{x} = \mathbf{\Psi w}, \tag{2.25}$$

where $\mathbf{\Psi} \in \mathbb{R}^{M \times L}$ maps the sparse domain to the acquisition domain and its l-th column is the l-th basis function, $\boldsymbol{\psi}_l \in \mathbb{R}^M$, evaluated at all M possible receivers. Therefore,

$$\mathbf{t} = \mathbf{\Omega}\mathbf{\Psi}\mathbf{w}. \tag{2.26}$$

Matrices with random numbers (Candes and Wakin, 2008) are often used for $\mathbf{\Omega}$ which correspond to the linear combination of the receivers with random coefficients. Nevertheless, such a choice limits the location of the receivers and is restrictive in the real world. Therefore, $\mathbf{\Omega}$ is set as the zero matrix, apart from one non-zero entry equal to 1 per row. This corresponds to a receiver at that location. $\mathbf{\Phi} = \mathbf{\Omega}\mathbf{\Psi}$ is used for simplicity and therefore

$$\mathbf{t} = \mathbf{\Phi}\mathbf{w}, \tag{2.27}$$

where $\mathbf{\Phi} \in \mathbb{R}^{N \times L}$. The l-th column of $\mathbf{\Phi}$ is the l-th basis function evaluated at only N receivers, denoted by $\boldsymbol{\phi}_l \in \mathbb{R}^N$. Variations to the formulation of equation 2.27 exist which insert zeros at the location of missing receivers (i.e. a row of zeros corresponds to no measurement taken) and operate in $\mathbb{R}^M$. The concept of the columns of the matrix being evaluated as basis functions is lost in this case. With the zeros inserted, it is a different basis function, but for the moment the discussion is continued with $\mathbf{\Omega}$.

One approach to solve this under-determined system is to set a sparsity constraint, by minimising the l_0 "norm" of $\mathbf{w}$. Note the quotation marks as this is not a proper norm but rather the number of non-zero elements of $\mathbf{w}$. If we define $0^0 = 0$, $\|\mathbf{w}\|_0$ is defined by

$$\|\mathbf{w}\|_0 = \sum_{l=1}^{L} |w_l|^0. \tag{2.28}$$

However, minimising the l_0 "norm" cannot be solved in polynomial time in general (Natarajan, 1995). The breakthrough in CS was made by a series of papers (Candes and Tao, 2006; Donoho, 2006) that enabled linear programming methods to find an approximate solution to the minimisation of the l_0 norm by minimising the l_1 norm using the following formulation

$$\hat{\mathbf{w}} = \min_{\mathbf{w}} \|\mathbf{w}\|_1 \quad \text{subject to} \quad \mathbf{\Phi}\mathbf{w} = \mathbf{t}. \tag{2.29}$$

Research in CS has been focused on algorithms that are able to solve l_1 minimisation problems, formally called the basis pursuit problem (Chen et al., 2001). Matching Pursuit (Mallat and Zhang, 1993) has been introduced that searches for the optimum basis functions in a greedy fashion to approximately obtain the sparsest solution. For

every iteration of the Matching Pursuit, a column of $\mathbf{\Phi}$ is chosen that gives the largest normalised inner product with the residual. The algorithm starts by initialising the residual, $\mathbf{r}^{(1)} = \mathbf{t}$, the coefficients, $\mathbf{w}^{(1)} = \mathbf{0}$ and $i = 1$. Then, it repeats

$$j = \arg \max_{l} \frac{\boldsymbol{\phi}_l^T \mathbf{r}^{(i)}}{\|\boldsymbol{\phi}_l\|_2} \quad l = 1, ..., L \tag{2.30}$$

where arg $\max_l$ means that the algorithm searches for the index, l, of the basis function that maximises the expression. Accordingly the j-th element of $\hat{\mathbf{w}}$ is updated by

$$\hat{w}_j^{(i+1)} = \hat{w}_j^{(i)} + \frac{\boldsymbol{\phi}_j^T \mathbf{r}^{(i)}}{\|\boldsymbol{\phi}_j\|_2^2} \tag{2.31}$$

where the residual is defined as

$$\mathbf{r}^{(i+1)} = \mathbf{r}^{(i)} - \boldsymbol{\phi}_j \frac{\boldsymbol{\phi}_j^T \mathbf{r}^{(i)}}{\|\boldsymbol{\phi}_j\|_2^2}. \tag{2.32}$$

This is repeated until the iterations are equal to the number of non-zero coefficients in the desired signal $\mathbf{w}$ or until the residual is sufficiently small. Variations of the above algorithm have been introduced such as the Orthogonal Matching Pursuit (OMP) (Tropp and Gilbert, 2007) which ensures that the residual is orthogonal to the already selected basis functions. In addition, hard thresholding (Blumensath and Davies, 2008) and soft thresholding (Daubechies et al., 2004) methods provide an alternative approach that uses knowledge of the sparsity of the desired signal to keep only the largest in magnitude elements at certain iterations. For further details and review, refer to Bryan and Leise (2013) and for more thorough descriptions of CS refer to Foucart and Rauhut (2013).

2.4 Dictionaries of basis functions and basis points

So far the discussion focused on CS algorithms but not yet on the dictionary of basis functions. An important distinction needs to be given between two types of bases that we will use in this thesis. First, basis functions such as the Haar wavelets are described by analytic expressions that specify their function. Hence the emphasis on basis *functions*. On the other hand, a dictionary of bases that is learnt by a feature learning algorithm (we will see examples in chapter 5) is composed of point values inferred on a grid with no underlying function. We will give a description of the three dictionaries of basis functions used by various CS algorithms in this thesis.

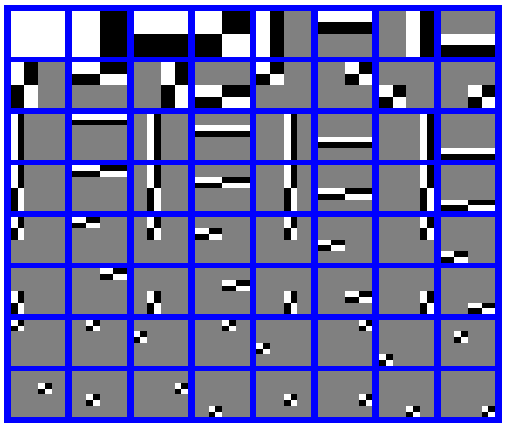


Fig. 2.15 The Haar wavelets transform.

Haar wavelets transform

The Haar wavelet is a popular type of transform due to its simplicity and fast calculation. It is composed of a base function defined as,

$$b(x) = \begin{cases} 1 & 0 \leq x < \frac{1}{2}, \\ -1 & \frac{1}{2} \leq x < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.33}$$

and its scaling function,

$$s(x) = \begin{cases} 1 & 0 \leq x < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.34}$$

The dictionary is composed of one scaling function and the rest are scaled and shifted versions of the base function. The scaled and shifted versions are governed by,

$$b_{n,k}(x) = 2^{\frac{n}{2}} b(2^n x - k), \tag{2.35}$$

where n controls the scale and k the shift over the entire input domain. Figure 2.15 shows the dictionary in an 8×8 domain where the respective 1D functions were multiplied together to obtain the 2D versions.

Discrete Cosine Transform

The Discrete Cosine Transform (DCT) is another popular transform that is used as a dictionary of basis functions. It is composed of a sum of cosine functions with different

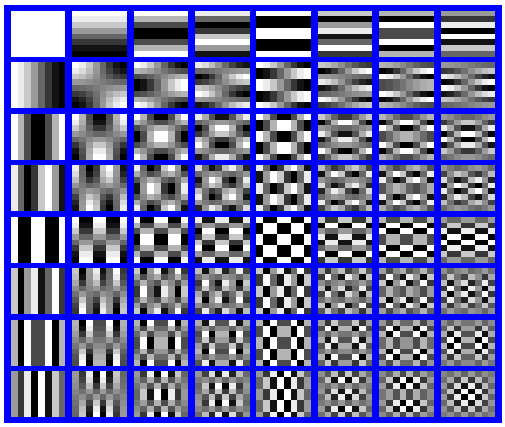


Fig. 2.16 The Discrete Cosine Transform.

frequencies. The DCT is related to the discrete Fourier Transform but uses only real numbers to represent the dictionary. It is defined by,

$$b(k) = \sum_{n=0}^{N-1} \cos\left[\frac{\pi}{N} k \left(n + \frac{1}{2}\right)\right] \tag{2.36}$$

for all input data points $k = 0, 1, ..., N-1$. Figure 2.16 illustrates the DCT on 8×8 space with different frequencies. Again, the respective 1D basis functions were multiplied together to obtain the 2D versions.

Gaussian basis functions

The Gaussian or radial basis functions are another commonly used dictionary defined by,

$$b(k, k') = \exp\left(-\frac{\|k - k'\|^2}{2\sigma^2}\right) \tag{2.37}$$

where k and k' span the location of the input space that the dictionary acts. σ^2 is a scale parameter to be set. This is similar to the exponential term of a normal (Gaussian) distribution. Figure 2.17 shows basis functions generated by this dictionary. It can be seen that each basis function looks like a small normal distribution that acts on different locations in the 8×8 space.

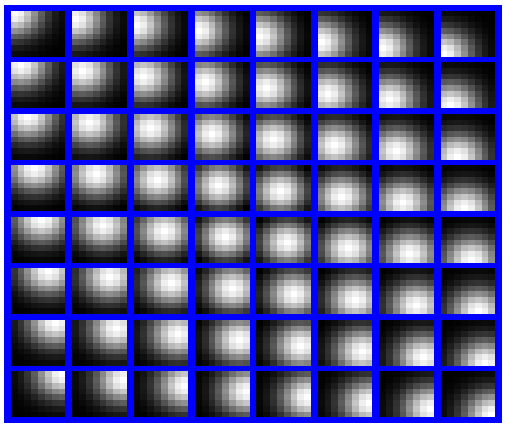


Fig. 2.17 The Gaussian basis functions.

2.5 An overview of seismic interpolation and Compressive Sensing

Each field has its own specific challenges and algorithms with solutions. In this section we will give descriptions of seismic Compressive Sensing (CS) methods and in general, an overview of seismic interpolation.

There are four main types of algorithms that solve the seismic interpolation problem: *prediction filters* that use non aliased low frequencies of seismic data to reconstruct the aliased parts (Naghizadeh and Sacchi, 2007; Porsani, 1999; Spitz, 1991), *wave equation solvers* that are based on wave-physics principles and require subsurface parameters (Ronen, 1987), *rank reduction solvers* that assume that missing receivers and noise increase the rank of the data (Gao et al., 2013; Kreimer and Sacchi, 2011, 2012; Kumar et al., 2015; Oropeza and Sacchi, 2011; Trickett et al., 2010) and *transform-based solvers* that use the assumption of sparsity for seismic data in a specific domain. Seismic CS falls into this last category of sparsity and techniques from this type will be used and explained further.

Seismic CS (Baraniuk and Steeghs, 2017) is treated as an inverse problem where seismic events are assumed to be sparse in some transform such as the Fourier (Abma and Kabir, 2006; Gülünay, 2003; Liu and Sacchi, 2004; Sacchi et al., 1998; Xu et al., 2005), the Radon (Kabir and Verschuur, 1995; Trad et al., 2002), the curvelet (Hennenfent and Herrmann, 2008; Herrmann and Hennenfent, 2008; Naghizadeh and Sacchi, 2010; Shahidi et al., 2013), the focal (Kutscha and Verschuur, 2016), the seislet (Fomel and Liu, 2010; Liu and Fomel, 2010) and the shearlet transform (Kong and Peng, 2015). These

dictionaries of basis functions are used in conjunction with a sparse solver to obtain a solution given the data.

In particular, Projection Onto Convex Sets (POCS) (Abma and Kabir, 2006) transforms the available data to various domains but more traditionally to the Fourier domain and uses hard or soft thresholding (Stanton et al., 2015) when choosing which components to keep in the solution. Various extensions exist for simultaneous source acquisition (Abma et al., 2015) and to work for under-sampled arbitrary irregular acquisition (Jiang et al., 2017). Another solver such as the Iteratively Reweighted Least Squares was also proposed (Zwartjes and Sacchi, 2007) to suppress the artifacts in the Fourier domain. For the curvelet transform, the Iterative Soft Thresholding (IST) (Herrmann and Hennenfent, 2008) was used. A faster version of IST was proposed (Beck and Teboulle, 2009), namely the Fast Iterative Soft Thresholding Algorithm (FISTA) and then applied to seismic data (Pérez et al., 2013). In a comparison of interpolators, POCS was found to preserve the amplitudes of seismic signals better (Stanton et al., 2012).

A different approach to thresholding is to solve the l_1-norm minimisation problem of equation 2.29. Spectral Projected Gradient for L1 (SPGL1) (van den Berg and Friedlander, 2009) was proposed to solve this and is used in the literature obtaining state-of-the-art results with various dictionaries (Jingjie et al., 2015; Kutscha and Verschuur, 2016). We will use POCS and SPGL1 in this thesis as a benchmark in comparisons. A brief description of each is given below.

Projection Onto Convex Sets

POCS (Abma and Kabir, 2006) is widely used in the field due to its effectiveness, simplicity and fast running time which allows it to scale to many dimensions. It inserts zeros at the location of missing receivers and then uses the Fast Fourier Transform (FFT) to transform the seismic data to the Fourier domain. This domain represents the Fourier coefficients and is usually sparse. To enforce sparsity further, a thresholding operator is used that removes low amplitudes in the coefficients of the Fourier transform. That is, only the largest coefficients are kept and then the Inverse Fast Fourier Transform (IFFT) is used to obtain an estimate of the signal. The number of iterations is defined at initialisation and at every iteration, the available data are used along with the new estimated values in order to calculate the next FFT. It is possible to use POCS with other basis functions, but in this thesis, we will use the original version with the Fourier Transform.

Spectral Projected Gradient for L1

Spectral Projected Gradient for L1 (SPGL1) (van den Berg and Friedlander, 2009) estimates the root of a non-linear equation with a single variable (see equation 2.39) instead of solving the problem in equation 2.29 directly. This variable, τ, is used in the definition of the LASSO problem (Tibshirani, 1994) defined by

$$\min_{\mathbf{w}} \|\mathbf{\Phi w} - \mathbf{t}\|_2 \text{ subject to } \|\mathbf{w}\|_1 \leq \tau. \tag{2.38}$$

Under certain conditions (van den Berg and Friedlander, 2009), problems 2.29 and 2.38 are identical and SPGL1 tries to estimate the variable τ that would give a minimiser for 2.29. First, an initial value $\tau = \tau_0$ is chosen (usually 0, unless a good estimate of τ is available). At each iteration k, Newton's method for root finding is used

$$\tau_{k+1} = \tau_k + \Delta\tau_k, \tag{2.39}$$

where $\Delta\tau_k = \sigma - \phi(\tau_k)/\phi'(\tau_k)$, $\phi(\tau) = \|\mathbf{r}_\tau\|_2$, $\mathbf{r}_\tau = \mathbf{t} - \mathbf{\Phi w}$ and σ is any assumed noise.

Therefore, for each iteration k, a LASSO problem (2.38) needs to be solved. This is done using the Spectral Projected Gradient (SPG) solver (Birgin et al., 2003) that returns $\mathbf{w}$ for a given τ. This is repeated until the convergence criteria are satisfied such as criteria on the residual and the number of iterations.

2.5.1 Seismic feature learning in Compressive Sensing

All the above algorithms utilise dictionaries of predefined basis functions for sparse representation. This limits the reconstruction to the assumption that every seismic signal, with any structure at any instance of operation, is sparse in the same transform as every other instance. This assumption does not allow for large signal variations and potential loss of reconstruction accuracy could occur. An alternative to the predefined dictionaries would be to learn the basis functions from the available seismic data.

This approach was used by Zhu et al. (2015) for the purpose of denoising seismic data with great success. The main algorithm is a modification of the K-Singular Value Decomposition (SVD) (Elad and Aharon, 2006) which alternates between optimising the coefficients and the dictionary, for the given data. Furthermore, simultaneous denoising and feature learning of seismic signals was performed by Beckouche and Ma (2014), and further dictionary learning for denoising was undertaken by Turquais et al. (2015). Another approach is to use a data-driven tight frame and learn a set of filters (features/bases) to sparsely represent seismic data (Liang et al., 2014) obtaining state-

of-the-art results similar to POCS. Yu et al. (2015) extended this for high-dimensional seismic data with great reconstruction accuracy but high computational cost. Yu et al. (2016) then used less patches during training by carefully selecting optimum patches depending on their variance in order to speed up the process with success. An alternative to speeding up the learning process using tight frames was studied by Siahsar et al. (2017) that use a non negativity constraint to reduce the space of the solution, consequently decreasing the computational cost and boost sparsity in data representation.

Another recent feature learning algorithm constrains the components to represent linear events of known slopes and using the slope information, the seismic data can be easily interpolated (Turquais et al., 2017). Furthermore, an Online Dictionary Learning (ODL) algorithm that processes one part of the training set at a time with stochastic approximations leads to faster performance instead of processing all at the same time as done in K-SVD (Tian et al., 2017). Issues of computation in feature learning are also addressed by Chen et al. (2016) with the use of a double-sparsity dictionary model to combine the fast fixed basis functions and the slow learning of features in a synthesis and analysis based model. Feature learning usage is growing in other aspects of seismic signal processing as well such as Full Waveform Inversion (FWI) (Zhu et al., 2017) and is important to understand its inner workings. We give a brief description next and refer the reader to chapter 5, for further technical details, where we will propose to use a new feature learning algorithm for seismic CS.

An introduction to feature learning

The choice of appropriate dictionary of basis functions, $\mathbf{\Psi}$ from equation 2.25, is fundamental for the solution of interpolation problems (Bengio et al., 2013). Researchers have been using their domain expertise to design suitable basis functions for their specific application and careful engineering is necessary to identify those that model the data well. Feature Learning is a set of algorithms that learn features/bases from raw data, deciding which are most suitable. In the context of CS, the task is to find a sparse representation for the training data. These can be bases at one common scale, or at multiple scales acquired through deep learning using many layers. In this thesis, we will focus on learning bases at one scale.

There are different routes to the solution, direct or indirect ones. Indirectly solving this problem involves methods that use available training data offline, learn the dictionary of bases and then use it for a desired task. Offline here means that the algorithms use a large collection of stored training data to learn bases. Such methods are the Denoising Autoencoders (Vincent et al., 2008), the Contractive Autoencoders (Rifai et al., 2011) and

the Restricted Boltzmann Machines (Hinton, 2002) to name a few. On the other hand, a direct way learns the dictionary of bases online, which in this case online means learning bases at the same time as interpolating/denoising the data using only the available receivers.

In order to achieve this, the signal $\mathbf{x} = \mathbf{\Psi w} \in \mathbb{R}^M$ is divided into T subsets $\mathbf{x}^{(i)}$, $i = 1, ..., T$ of size $K = M/T$. For example, if we want to learn a dictionary of bases for a two-dimensional signal of size 128×128, that is $M = 16384$, we can split the signal into $T = 256$ patches of size 8×8, that is $K = 64$. Patches are usually extracted with overlaps to increase the number of training subsets (refer to section 5.2 for details). We will introduce a new variable, $\mathbf{D} \in \mathbb{R}^{K \times L}$, which represents the dictionary of bases for learning. It is assumed that each training subset arises from a vector of coefficients, $\mathbf{w}^{(i)}$, in the sparse domain under the same dictionary, $\mathbf{D}$, with additive noise, $\boldsymbol{\epsilon}^{(i)}$, given by,

$$\mathbf{x}^{(i)} = \mathbf{D}\mathbf{w}^{(i)} + \boldsymbol{\epsilon}^{(i)}. \tag{2.40}$$

Let $\mathbf{X} \in \mathbb{R}^{K \times T}$ be the matrix with columns $\mathbf{x}^{(i)}$, $i = 1, ..., T$ and $\mathbf{W} \in \mathbb{R}^{L \times T}$ with columns $\mathbf{w}^{(i)}$, $i = 1, ..., T$. The goal is to infer simultaneously $\mathbf{D}$ and $\{\mathbf{w}^{(i)}\}_{i=1}^T$ from the signal subsets $\{\mathbf{x}^{(i)}\}_{i=1}^T$ via the optimisation problem

$$\min_{\mathbf{D},\mathbf{W}} \|\mathbf{X} - \mathbf{DW}\|_2^2 \quad \text{subject to} \quad \|\mathbf{w}^{(i)}\|_0 \leq T_0, \quad \text{for } i = 1, ..., T \tag{2.41}$$

where $T_0 \ll K$ is the sparsity (number of non-zero elements) of the signal and is usually set empirically beforehand. This is done in K-SVD (Aharon et al., 2006), which alternates between optimising $\mathbf{D}$ and $\mathbf{W}$ and uses a pursuit algorithm to compute the coefficients $\mathbf{w}^{(i)}$ for each training subset $\mathbf{x}^{(i)}$.

Note that in equation 2.41 no sensing matrix $\mathbf{\Omega}$ is employed, which would possibly be different for each subset. Instead of using $\mathbf{t}^{(i)} = \mathbf{\Omega}^{(i)}\mathbf{x}^{(i)}$, the components of $\mathbf{x}^{(i)}$ where data is missing are set to zero (Aharon et al., 2006). In the case where no data is missing but rather noise is present, the original values with noise are used (Elad and Aharon, 2006). A mask is used to indicate the locations of available data. Inserting zeros in the place of missing data points is also done in POCS which helps preserve the location and structure inside each $\mathbf{x}^{(i)}$. However, SPGL1 employs a sensing matrix $\mathbf{\Omega}$ and collapses the data as in equation 2.27 which is the traditional CS formulation (Candes and Wakin, 2008).

Learning the dictionary of bases at the same time as performing denoising and/or interpolation uses training data that are corrupted. One might expect that the learned dictionary is only useful for sparsely representing the corrupted signals. However, this

is not the case as examined in various models for feature learning (Srivastava et al., 2014; Vincent et al., 2008) and in seismic applications (Beckouche and Ma, 2014) where denoising and feature learning were performed simultaneously. In fact, adding noise or dropping out measurements from the training data is recommended as a regularisation in order to avoid over fitting (Bengio et al., 2013). Furthermore, employing many training subsets mitigates the risk of learning corruption.

In this thesis, we propose to use a feature learning algorithm called Beta Process Factor Analysis (BPFA) that uses the above principles but at the same time it is built around a Bayesian framework of machine learning. In the next subsection, we give other examples of machine learning and Bayesian statistics algorithms used in seismic data processing.

2.5.2 Machine learning and Bayesian statistics for seismic applications

Recently, a machine learning technique called the Support Vector Regression (SVR) (Jia and Ma, 2017) has been used for seismic interpolation with success by learning a hyper-plane that describes the relationship between input and output data. It defines and learns a function that maps the inputs to the outputs of the training data and then it is able to generalise for unseen receivers with great success. A faster version has been proposed by Jia et al. (2018) which uses a subset of the training data using Monte Carlo. SVR is a standard supervised learning algorithm with its general form applied for classification, namely the widely known Support Vector Machines (SVM) (Cortes and Vapnik, 1995).

The SVR and all other algorithms discussed in this chapter are deterministic. They predict (or interpolate) the receivers' values but without associating an uncertainty or confidence about the values. As discussed before, an uncertainty map would be advantageous in order to associate risk or to provide information about future seismic survey designs. Thus, in order to tackle this, probabilistic seismic Compressive Sensing (CS) algorithms can help.

Bayesian statistics provide a probabilistic framework for this. Excellent introduction to Bayesian statistics for seismic data is given by Duijndam (1988a,b) and Ulrych et al. (2001). Malinverno and Briggs (2004) expanded this using empirical Bayes for uncertainty quantification. Other applications of Bayesian estimation can be found by Wang et al. (2008) for seismic wavefield separation, for estimating model uncertainties in FWI (Zhu et al., 2016) and for petrophysics-seismic inversion (Fjeldstad and Grana, 2018).

Bayesian Compressive Sensing (CS) (Ji et al., 2008) provides the solution to the problem of quantifying uncertainty of predicted receiver values. We will build on this for seismic CS in order to create data-driven models for seismic data. A probabilistic version of the SVM (and the SVR) is the Relevance Vector Machine (RVM) that we mentioned earlier and we will use it in this thesis (refer to section 4.1 for further information). To facilitate the discussion for the RVM and subsequently for the BPFA, we will provide an introduction to machine learning and Bayesian statistics in the next chapter.

Chapter 3

Machine Learning and Bayesian Statistics

Using data-driven models to describe real-world observations has increased in popularity in recent years. Uncertainty is a core component of both the model and the measurements and models that are able to capture it are very desirable. Bayesian statistics in machine learning is a framework that tackles this by allowing the construction of probabilistic data-driven models using probability distributions. By using appropriate assumptions about the generative process of each variable, it is possible to create accurate probabilistic models to solve various tasks. In this thesis, we will work with two types of probabilistic models. Before moving into the specific details, we will describe the general principle of Bayesian modelling.

3.1 Introduction to Bayesian modelling

Probabilistic models are usually composed of latent variables and model parameters. A latent variable is an unobserved variable of the model that we would like to infer from the data. On the other hand, a parameter can be either set or inferred. There is a significant difference between the two in that for every data point, there is a corresponding latent variable as opposed to fixed-size model parameters. Thus, the number of latent variables grows with the number of data points as opposed to a fixed number of pre-defined model parameters.

Figure 3.1 shows a graphical model of this concept, where $\mathbf{x}^{(i)}$ is an observed data point and $\mathbf{z}^{(i)}$ is the latent variable for this model. The plate indicates that there is one variable per data point. θ is a model parameter that characterises the distribution of the latent variables (i.e. if we assume a normal distribution, it can be its mean). In this

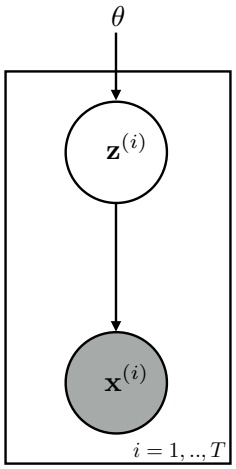


Fig. 3.1 Graphical model with latent variables and model parameter.

thesis, we will work with a latent variable model (more specifically, latent feature model) and a model with fixed parameters.

In order to facilitate the discussion on modelling random variables with probability distributions, three definitions are provided. Given two random variables $\mathbf{x} \in \mathbb{R}^K$ and $\mathbf{z} \in \mathbb{R}^L$ where K and L are arbitrary dimensions,

- the joint probability distribution $p(\mathbf{x}, \mathbf{z})$ is the probability distribution of both random variables having a certain configuration simultaneously.
- the conditional probability distribution $p(\mathbf{x}|\mathbf{z})$ is the probability distribution of $\mathbf{x}$ given that $\mathbf{z}$ is known.
- the marginal probability distribution $p(\mathbf{x})$ is the probability distribution for a specific configuration of $\mathbf{x}$. It is called marginal because it can be obtained by marginalising (or integrating) other random variables out.

Bayesian modelling constructs a model using two fundamental rules of probability theory, the sum rule and the product rule. The sum rule states that the marginal distribution of one variable is equivalent to marginalising the joint distribution of two variables over the second. That is,

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) d\mathbf{z}, \tag{3.1}$$

where the integral is performed over all possible configurations of $\mathbf{z}$. The product rule on the other hand states that the joint distribution between random variables is obtained by the product of the conditional distribution of one given the other and the marginal

distribution of the other. That is

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}). \tag{3.2}$$

The same applies to $p(\mathbf{z}, \mathbf{x})$ which is equivalent to $p(\mathbf{x}, \mathbf{z})$ by symmetry. Thus,

$$p(\mathbf{z}|\mathbf{x})p(\mathbf{x}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}). \tag{3.3}$$

The Bayes' rule is obtained by re-arranging the above giving

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x})} \quad = \quad \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{\int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) d\mathbf{z}}. \tag{3.4}$$

Using the Bayes' rule, it is possible to use data to infer an underlying model along with uncertainty information. Reconsider, $\mathbf{x} \in \mathbb{R}^K$ as the data vector and $\mathbf{z} \in \mathbb{R}^L$ as the latent variables of the model. The two conditional probability distributions have special names along with the marginal,

- $p(\mathbf{z}|\mathbf{x})$ is the posterior probability distribution of the latent variables, $\mathbf{z}$, given the data, $\mathbf{x}$. A significant effort in Bayesian modelling involves learning this distribution from the available data.

- $p(\mathbf{x}|\mathbf{z})$ is the likelihood function and can also be referred as the model of the data. Another way to think about this would be the probability of a particular configuration of the model variables, $\mathbf{z}$, producing the data, $\mathbf{x}$. A model configuration describing the data well will have large likelihood, while a poor model will have low likelihood.

- $p(\mathbf{z})$ is the prior distribution for the model variables, $\mathbf{z}$, and is chosen according to any prior beliefs about the model to be constructed.

The complete specification of a Bayesian model is thus given by the joint probability distribution of all variables and parameters. For example, for Figure 3.1, the joint distribution of the data, $\mathbf{x}$, the latent variables, $\mathbf{z}$, and the model parameter, θ, is given by $p(\mathbf{x}, \mathbf{z}, \theta)$. The aim is to subsequently learn the latent variables and unknown model parameters. Nevertheless, there are challenges to achieve it. Appropriate likelihood functions and prior distributions need to be chosen. In some cases these can lead to complicated expressions. Most importantly, the integral in the denominator of equation 3.4 could be intractable, since it has to consider all possible configurations of the model variables, $\mathbf{z}$, which could be defined in a high-dimensional space. The essence in Bayesian

modelling is how to use assumptions about the data to choose appropriate distributions and to choose inference algorithms that can learn the posterior distribution of the model variables. One popular class of inference algorithms that approximate posterior distributions via sampling is described next and will be used in this thesis.

3.2 Sampling with Markov Chain Monte Carlo

First-order Markov Chain Monte Carlo (MCMC) is used when it is not possible to draw samples from a probability distribution but rather only able to evaluate it upto a normalising constant. Before discussing the sampling scheme, an introduction to Markov Chains will be given to motivate MCMC.

A Markov Chain is a stochastic process in which future states are independent of past states given the current state. Consider a random variable $\mathbf{z} \in \mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_s\}$ where s is the number of possible configurations (states) it can take. A draw (sample) of $\mathbf{z}^t$ from its corresponding distribution is a state at iteration t. The next draw is dependent only by the current draw $\mathbf{z}^t$ and not on any of the past draws resulting in the following Markov property

$$p(\mathbf{z}^{t+1}|\mathbf{z}^t, \mathbf{z}^{t-1}, ..., \mathbf{z}^1) = p(\mathbf{z}^{t+1}|\mathbf{z}^t). \tag{3.5}$$

That is, the Markov chain is a set of samples that are each slightly dependent on the previous one. The chain moves around the variable's space and remembers only where it was in the previous state. As t approaches infinity, under certain conditions, the samples are drawn from the desired distribution called the invariant distribution, for any starting distribution. This stability is the key to MCMC since if the invariant distribution is the target distribution then it is possible to sample from it by initialising the chain from any point in the variable's space. This is satisfied however under certain conditions of irreducibility and aperiodicity. Irreducibility means that for any state of the chain, there is positive probability of visiting all other states and aperiodicity means that the chain is not trapped in cycles. Thus, MCMC are a class of methods that simulate draws that are slightly dependent but can be used to approximate a target distribution. Two very popular methods are discussed below where the latter can be seen as a special case of the former and will be used in this thesis.

3.2.1 Metropolis-Hastings

The Metropolis-Hastings (MH) algorithm is the most popular MCMC sampler and many practical MCMC samplers can be interpreted as special cases of MH. It requires a proposal

distribution $q(\mathbf{z})$ and involves the sampling of a candidate sample (state) $\mathbf{z}^*$ given the current state. That is, a new sample is drawn from $q(\mathbf{z}^*|\mathbf{z}^t)$ given the current state. This sample however is not guaranteed to be accepted. The Markov chain only moves to $\mathbf{z}^*$ if a condition is satisfied, otherwise it stays at its current state and tries a new sample. A draw from a uniform distribution is obtained in the interval $[0, 1]$ and then compared with $\mathcal{A}(\mathbf{z}^t, \mathbf{z}^*)$ which is defined by

$$\mathcal{A}(\mathbf{z}^t, \mathbf{z}^*) = \min\left\{1, \frac{p(\mathbf{z}^*)q(\mathbf{z}^t|\mathbf{z}^*)}{p(\mathbf{z}^t)q(\mathbf{z}^*|\mathbf{z}^t)}\right\}. \tag{3.6}$$

This means that a new sample is accepted with probability $\mathcal{A}(\mathbf{z}^t, \mathbf{z}^*)$ and heavily depends on the choice of the proposal distribution. It can be shown (Andrieu et al., 2003) that the samples generated by the MH algorithm are draws from the target distribution. The algorithm always allows for rejection and therefore satisfies the aperiodic condition. For irreducibility, the support of the proposal distribution has to include the support of the target distribution. Thus, great care is needed when proposing $q(\mathbf{z})$. A summary of the algorithm can be seen in Algorithm 1. Note that it is only necessary to know $p(\mathbf{z})$ up to a constant of proportionality since it appears in the form of ratios and the constants cancel each other out.

Algorithm 1 Metropolis-Hastings Algorithm

Require: Initialise $\mathbf{z}^0$ and number of samples N

for $t = 0$ to $N - 1$ **do**
  $u \sim \mathcal{U}_{(0,1)}$
  $\mathbf{z}^* \sim q(\mathbf{z}^*|\mathbf{z}^t)$
  if $\left(u < \mathcal{A}(\mathbf{z}^t, \mathbf{z}^*) = \min\left\{1, \frac{p(\mathbf{z}^*)q(\mathbf{z}^t|\mathbf{z}^*)}{p(\mathbf{z}^t)q(\mathbf{z}^*|\mathbf{z}^t)}\right\}\right)$
    $\mathbf{z}^{t+1} = \mathbf{z}^*$
  else
    $\mathbf{z}^{t+1} = \mathbf{z}^t$
return $(\{\mathbf{z}^t\}_{t=0}^{N-1})$

The specific choice of the proposed distribution can change the performance of the algorithm. It is very common to use a normal distribution as $q(\mathbf{z}^*|\mathbf{z}^t) = \mathcal{N}(\mathbf{z}^t, \sigma^2)$ which is easy to sample and the results can be easily interpreted. Different choices of the proposal standard deviation, σ, lead to very different results. If the standard deviation is too small, the acceptance rate will be high but with poor performance since only a small proportion of probability space will be covered and thus many modes of probability mass could be missed. On the other hand, if the standard deviation is too big, the rejection

rate can be very high because of the spaces visited for which the target probability is very small. The chain will not move and the samples will be highly correlated. If all the modes and large proportion of the probability space is explored with high acceptance rate, then the MH sampler has good sampling performance with the chain mixing well.

3.2.2 Gibbs Sampling

Gibbs Sampling (Geman and Geman, 1984) is a special case of the MH algorithm which is widely used due to its simplicity. Consider $\mathbf{z} \in \mathbb{R}^L$ and that the full conditionals $p(z_j|z_1, z_2, ..., z_{j-1}, z_{j+1}, ..., z_L)\ \forall j$ are available. Gibbs sampling uses as a proposal distribution for $j = 1, .., L$

$$q(\mathbf{z}^*|\mathbf{z}^t) = \begin{cases} p(z_j^*|\mathbf{z}_{-j}^t) & \text{if } \mathbf{z}_{-j}^* = \mathbf{z}_{-j}^t \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

where $\mathbf{z}_{-j}^t = [z_1^t, z_2^t, ..., z_L^t]$ with j omitted. On the other hand, $\mathbf{z}_{-j}^* = [z_1^*, z_2^*, ..., z_L^*]$ with j omitted. The corresponding acceptance probability is given by

$$\begin{aligned} \mathcal{A}(\mathbf{z}^t, \mathbf{z}^*) &= \min\left\{1, \frac{p(\mathbf{z}^*)q(\mathbf{z}^t|\mathbf{z}^*)}{p(\mathbf{z}^t)q(\mathbf{z}^*|\mathbf{z}^t)}\right\} \\ &= \min\left\{1, \frac{p(\mathbf{z}^*)p(z_j^t|\mathbf{z}_{-j}^*)}{p(\mathbf{z}^t)p(z_j^*|\mathbf{z}_{-j}^t)}\right\} \\ &= \min\left\{1, \frac{p(z_j^*|\mathbf{z}_{-j}^*)p(\mathbf{z}_{-j}^*)p(z_j^t|\mathbf{z}_{-j}^*)}{p(z_j^t|\mathbf{z}_{-j}^t)p(\mathbf{z}_{-j}^t)p(z_j^*|\mathbf{z}_{-j}^t)}\right\} \\ &= \min\left\{1, \frac{p(\mathbf{z}_{-j}^*)}{p(\mathbf{z}_{-j}^t)}\right\} \\ &= 1, \end{aligned} \tag{3.8}$$

where the fact that $\mathbf{z}_{-j}^* = \mathbf{z}_{-j}^t$ is used due to the fact that all these components of $\mathbf{z}$ are fixed/unchanged by that sampling iteration. In addition, $p(\mathbf{z}^*) = p(z_j^*|\mathbf{z}_{-j}^*)p(\mathbf{z}_{-j}^*)$ is used in order to simplify the expression. This means that the acceptance probability for each proposal is one and thus at every iteration, the full conditionals can be used to draw samples for the Markov Chain without throwing away any samples. The number of samples needs to be specified beforehand and matches the number of iterations of the sampler. It is very important to monitor the evolution of the sampler and make sure that the number of iterations is sufficient (refer to section 5.9 for an example of such an evolution). A summary of the algorithm can be seen in Algorithm 2.

Algorithm 2 Gibbs Sampling

Require: Initialise $\mathbf{z}^0 \in \mathbb{R}^L$ and number of samples N

- **for** $t = 0$ to $N - 1$ **do**
- $z_1^{t+1} \sim p(z_1 | z_2^t, z_3^t, ..., z_L^t)$
- $z_2^{t+1} \sim p(z_2 | z_1^{t+1}, z_3^t, ..., z_L^t)$
- $\vdots$
- $z_j^{t+1} \sim p(z_j | z_1^{t+1}, z_2^{t+1}, ..., z_{j-1}^{t+1}, z_{j+1}^t, ..., z_L^t)$
- $\vdots$
- $z_L^{t+1} \sim p(z_L | z_1^{t+1}, z_2^{t+1}, ..., z_{L-1}^{t+1})$

return $(\{\mathbf{z}^t\}_{t=0}^{N-1})$

In order to be able to use Gibbs sampling, we need the full conditional distribution of all latent variables and model parameters. Once we obtain these, we can repeatedly sample from the respective distributions. Many machine learning models use conjugate pairs of distributions (the prior and posterior distributions are in the same family) and consequently, we can obtain analytically in closed-form full conditional distributions that we can sample from directly. In cases where it is not possible to obtain known distributions, Metropolis-Hastings can be used. In the following section, we will provide popular probability distributions that will be used throughout the thesis.

3.3 Probability distributions and conjugacy

Different modelling assumptions about variables require different governing probability distributions. Throughout the thesis, we will be using various distributions and brief descriptions are given below.

Normal distribution

A random variable, x, is distributed by a normal distribution when,

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{3.9}$$

where $x \in \mathbb{R}$, μ is the mean of the distribution and σ^2 its variance. This distribution is very popular because it is generally used to model real-valued random variables in

many settings. This is due to the central limit theorem which states that under certain conditions, the average of multiple random variables which itself is a random variable has a distribution that tends to a normal in the limit. One example is the measurement error which is assumed to be the average of many other variables (such as equipment error, human error, etc).

Bernoulli distribution

A random variable, z, is distributed by a Bernoulli distribution when,

$$p(z;\pi) = \pi^z(1-\pi)^{(1-z)}, \tag{3.10}$$

where $z \in \{0,1\}$ and π is the probability of $z = 1$. This distribution can be used in situations where something either happens or it does not such as flipping a coin (heads/tails) or for a variable that states whether a data point has a feature or not.

Gamma distribution

A random variable, γ, is distributed by a Gamma distribution when,

$$p(\gamma;\alpha,\beta) = \frac{\beta^\alpha \gamma^{(\alpha-1)} e^{-\beta\gamma}}{\Gamma(\alpha)}, \tag{3.11}$$

where $\gamma > 0$. $\alpha, \beta > 0$ are the shape and rate parameters respectively with different settings changing the overall distribution. $\Gamma(\alpha) = \int_0^\infty x^{(\alpha-1)} e^{-x} dx$ is the Gamma function and is used to normalise the distribution. This distribution can be used to model positive scale parameters and one of its most useful applications is to model the precision (inverse of variance) of a normal distribution.

Beta distribution

A random variable, π, is distributed by a Beta distribution when,

$$p(\pi;\alpha,\beta) = \frac{1}{B(\alpha,\beta)} \pi^{(\alpha-1)}(1-\pi)^{(\beta-1)}, \tag{3.12}$$

where $\pi \in [0,1]$, $B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ is the Beta function and $\alpha, \beta > 0$. α and β are shape parameters that can change the shape of the distribution promoting different outcomes depending on their settings. This distribution can be used for a continuous variable

between 0 and 1. It is often used to model probabilities or proportions, for example the probability of obtaining head or tails when flipping a coin.

Conjugate pairs of distributions

As we discussed in the previous section, it is easier to sample from closed-form known distributions for the purposes of Gibbs Sampling. In order to obtain these, conjugate priors help simplify the derivations. A prior distribution is conjugate to a likelihood function if the posterior and prior distribution are in the same family. A concise list of conjugate pairs of distributions[1] is useful when defining new models.

3.4 Latent variable models

The probability distributions described in the previous section might not be suitable when dealing with complicated data. If the data are generated from multiple processes, one probability distribution with simple assumptions will not be able to capture the variations in the data. In order to tackle problems with multiple sources that correspond to complicated distributions, mixtures of variables from simpler probability distributions are used. These models are called latent variable models and two types are: latent class and latent feature models. In this thesis, we will use a latent feature model but to facilitate its description, we will first explain a latent class model and then build on that.

3.4.1 Latent class models

A particular model of interest is the Gaussian mixture model which as the name suggests, normal (or Gaussian) distributions are linearly combined to represent more complicated probability distributions. The problem of clustering can be solved with these models by modelling each cluster as a separate normal distribution. In order to illustrate the significance of Gaussian mixture models, consider a one-dimensional variable x distributed by $p(x)$ in Figure 3.2. This distribution can not be represented explicitly by a single expression. However, if it is possible to use a linear combination of different probability distributions to represent it, it is possible to obtain an equivalent and simpler representation. In this example, the particular distribution can be represented as a mixture of three normal distributions with different means and variances as it can be seen in Figure 3.3.

[1] Probability and Statistics cookbook, http://statistics.zone/ last accessed 8th March 2018

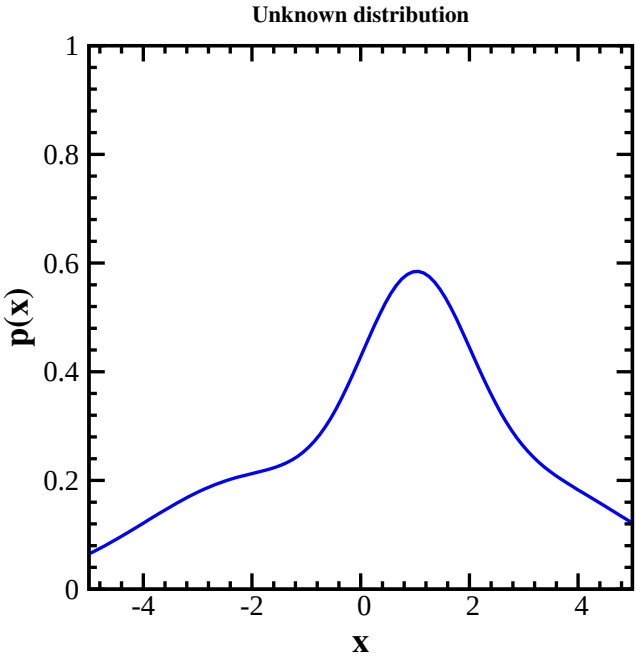


Fig. 3.2 Complicated distribution of a single variable.

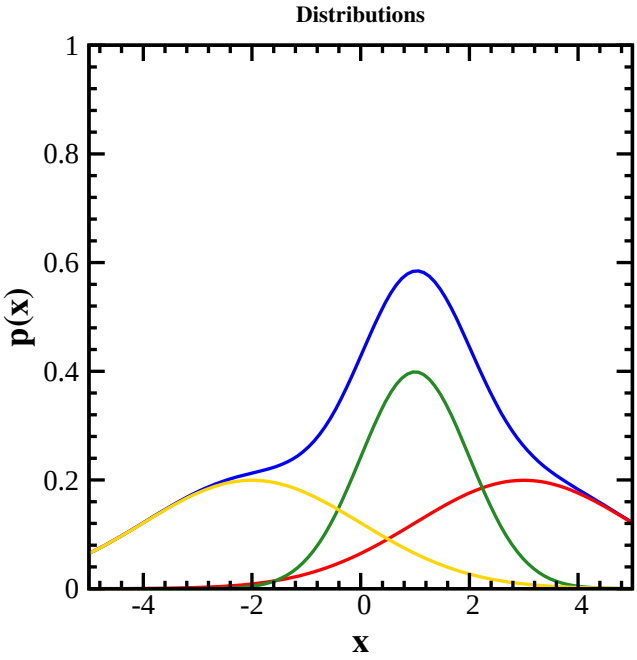


Fig. 3.3 Mixture of three normal distributions representing a complicated distribution.

In order to use Gaussian mixture models, the introduction of latent class variables is necessary. This is used in order to identify the contribution of each of the normal distributions (or clusters in clustering) out of all distributions (clusters) in the mixture. The latent class variables control the assignment of samples to base distributions. In order to behave as required, we introduce $\mathbf{z}$ having a categorical distribution. This is obtained by defining it as a binary random variable having a 1-of-L representation, belonging to 1-of-L classes, hence the name of latent class models. Thus, $\mathbf{z} \in \{0,1\}^L$ where L is the number of base distributions/clusters in the mixture. Due to the fact that this is a categorical model, $\mathbf{z}$ must satisfy

$$\sum_{l=1}^{L} z_l = 1, \tag{3.13}$$

which translates to only one non-zero element in $\mathbf{z}$. This vector of latent variables provides the mechanism to identify one distribution/cluster but it is necessary to have a probability of a sample belonging to any of the distributions/clusters. This is achieved by defining the marginal distribution of $p(\mathbf{z})$ as

$$p(z_l = 1) = \pi_l. \tag{3.14}$$

Since π_l are probabilities, they must satisfy $0 \leq \pi_l \leq 1$ and $\sum_{l=1}^{L} \pi_l = 1$. To obtain the marginal distribution over $\mathbf{z}$, recall that $\mathbf{z}$ is a 1-of-L vector and thus

$$p(\mathbf{z}) = \prod_{l=1}^{L} \pi_l^{z_l}. \tag{3.15}$$

This means that only π_l will be active, corresponding to the non-zero element in z_l (i.e. for the l-th distribution/cluster). However, by using $p(z_l = 1)$ for all l, it is possible to obtain the probability for every single distribution/cluster via the Bayes' rule as discussed later on.

Another vital component for this model is the conditional distribution of $\mathbf{x}$ given $\mathbf{z}$. If a sample is generated by a distribution l, then the conditional probability distribution is given by

$$p(\mathbf{x}|z_l = 1) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l), \tag{3.16}$$

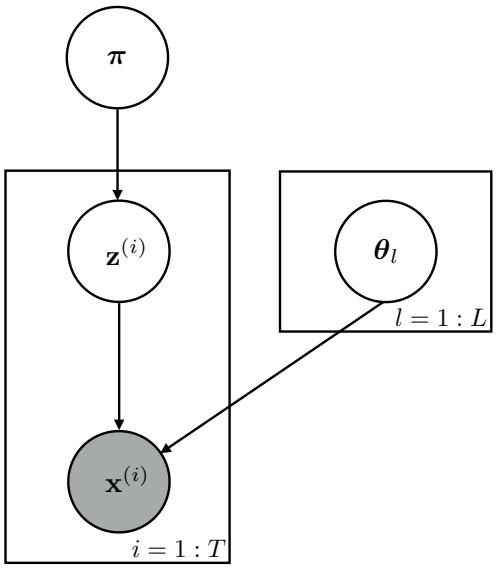


Fig. 3.4 Graphical model of the Gaussian mixture model.

where $\boldsymbol{\mu}_l$ is the mean of the l-th normal distribution and $\boldsymbol{\Sigma}_l$ its covariance. Consequently, the conditional distribution over all $\mathbf{z}$ is

$$p(\mathbf{x}|\mathbf{z}) = \prod_{l=1}^{L} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)^{z_l}. \tag{3.17}$$

Using the sum and product rules of probability theory, the marginal distribution of $\mathbf{x}$, which is the model of the samples, is given by

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) \\ &= \sum_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{x}|\mathbf{z}) = \sum_{l=1}^{L} \pi_l \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l). \end{aligned} \tag{3.18}$$

This model for $\mathbf{x}$ can be generalised for samples that are independent and identically distributed (i.i.d.), $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(T)}\}$ and for every sample there is a corresponding $\mathbf{z}^{(i)}$. The corresponding graphical model is shown in Figure 3.4 where $\boldsymbol{\theta}_l = \{\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\}$. Thus, the model is trained using a set of observations for clustering and indicates their distribution/cluster with a certain probability. The probability is obtained for all distribution/clusters in the mixture and is viewed as the responsibility that cluster l

takes for the explanation of sample $\mathbf{x}$. This is defined as

$$\begin{aligned}\gamma(z_l) &= p(z_l = 1|\mathbf{x}) \\ &= \frac{p(z_l = 1)p(\mathbf{x}|z_l = 1)}{p(\mathbf{x})} \\ &= \frac{p(z_l = 1)p(\mathbf{x}|z_l = 1)}{\sum_{j=1}^{L} p(z_j = 1)p(\mathbf{x}|z_j = 1)} = \frac{\pi_l \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}{\sum_{j=1}^{L} \pi_j \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)},\end{aligned} \tag{3.19}$$

where the Bayes' rule was used to obtain the posterior distribution. This is done for every distribution/cluster in the mixture to obtain the probability that sample $\mathbf{x}$ came from the l-th component.

Expectation Maximisation

Everything is now in place with respect to modelling and thus the discussion will continue on how to obtain the necessary statistics for the model given the training data (observations). A very powerful algorithm for training latent variable models is the Expectation Maximisation (EM) algorithm and follows a two-step iterative procedure similar to Gibbs sampling. Given observations $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(T)}\} \in \mathbb{R}^K$, it is beneficial to compactly represent them in $\mathbf{X} \in \mathbb{R}^{T \times K}$. Then, in order to obtain the relevant statistics $\{\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l, \pi_l\}_{l=1}^{L}$ from the observations, we maximise the marginal likelihood of the model. This is given by (3.18) but since there are T i.i.d. samples, its product is

$$p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{T} \sum_{l=1}^{L} \pi_l \mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l). \tag{3.20}$$

Taking the natural logarithm of the above expression for ease of calculation, the log-likelihood is given by

$$\ln\, p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{T} \ln \left\{ \sum_{l=1}^{L} \pi_l \mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right\}. \tag{3.21}$$

In order to obtain the maximum likelihood estimation with the corresponding statistics, the derivatives of the above are required. First, setting the derivative of (3.21) to zero with respect to $\boldsymbol{\mu}_l$

$$\sum_{i=1}^{T} \frac{\pi_l \mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}{\sum_{j=1}^{L} \pi_j \mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_l^{-1} (\mathbf{x}^{(i)} - \boldsymbol{\mu}_l) = 0. \tag{3.22}$$

Multiplying by $\boldsymbol{\Sigma}_l$ and re-arranging

$$\boldsymbol{\mu}_l = \frac{1}{N_l} \sum_{i=1}^{T} \gamma_l^{(i)} \mathbf{x}^{(i)}, \tag{3.23}$$

where $N_l = \sum_{i=1}^{T} \gamma_l^{(i)}$ translates to the number of points assigned to distribution/cluster l. It can be seen that the mean $\boldsymbol{\mu}_l$ is estimated by a weighted sum of all the samples with weights being their respective responsibilities. By using the above observation, the covariance is given by

$$\boldsymbol{\Sigma}_l = \frac{1}{N_l} \sum_{i=1}^{T} \gamma_l^{(i)} (\mathbf{x}^{(i)} - \boldsymbol{\mu}_l)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_l)^T, \tag{3.24}$$

which is a weighted version of the maximum likelihood estimate for a single normal distribution. Finally, the derivative with respect to π_l is required. Here, there is an additional constraint to satisfy the definition of probabilities and thus a Lagrange multiplier is added to convert the expression to an unconstrained optimisation resulting in

$$\ln\, p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \lambda \left(\sum_{l=1}^{L} \pi_l - 1 \right), \tag{3.25}$$

where λ is the Lagrange multiplier. Differentiating the above with respect to π_l and setting to zero

$$\sum_{i=1}^{T} \frac{\mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}{\sum_{j=1}^{L} \pi_j \mathcal{N}(\mathbf{x}^{(i)}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda = 0. \tag{3.26}$$

After some manipulations and using the constraint on the π_l

$$\pi_l = \frac{N_l}{T}. \tag{3.27}$$

From the above derivations, it can be seen that the statistics to be estimated depend on the responsibilities $\gamma_l^{(i)}$ which in turn depend on the statistics. Thus, an iterative scheme emerges for finding the solution to this maximisation problem of the likelihood. This procedure is the EM algorithm. It comprises of the Expectation step where the responsibilities are updated using the current state of the statistics. Then, the Maximisation step follows where, using the new responsibilities and the old statistics, new estimates for all three variables are obtained.

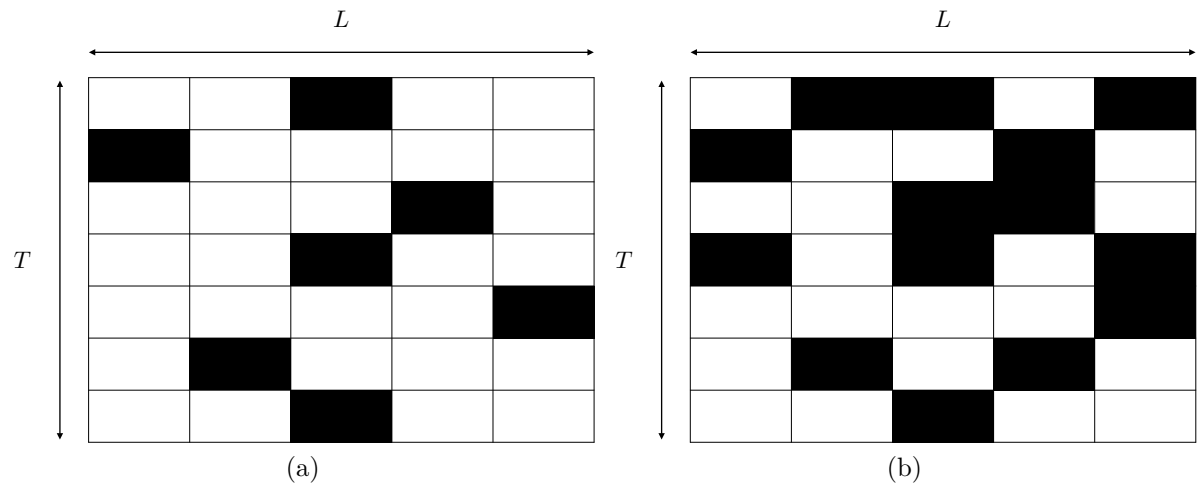


Fig. 3.5 Binary matrix with the restriction of only one non-zero element per data point (a). Latent feature models do not use this restriction by allowing multiple non-zero elements (features) per data point (b).

3.4.2 Latent feature models

Although there are applications that latent class models are useful, the assumption that a data point only belongs to one-of-L classes is very limiting. Real world data are more complex and a more complex representation power is essential. Figure 3.5(a) shows an illustration of the binary matrix $\mathbf{Z} \in \{0,1\}^{T\times L}$ for the latent class modelling. Only one non-zero element per data point is permitted. Latent feature models tackle this limitation by modelling a data point as a combination of many classes or rather features. That is, an observation can be modelled as a linear combination of many features (i.e. a human can be tall, an athlete and a writer) which are not mutually exclusive as opposed to the latent class models. Figure 3.5(b) shows an illustration of the binary matrix, $\mathbf{Z}$, for the latent feature model.

A classic model of this type is called factor analysis (Ghahramani and Hinton, 1997) where the data points are assumed to be generated by,

$$\mathbf{x}^{(i)} = \mathbf{D}\mathbf{w}^{(i)}, \tag{3.28}$$

all contained in their respective matrices, $\mathbf{X} = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(T)}\right]^T \in \mathbb{R}^{T\times K}$, the latent factors (coefficients) $\mathbf{W} = \left[\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, ..., \mathbf{w}^{(T)}\right]^T \in \mathbb{R}^{T\times L}$ and the factor loadings (features or bases) $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_L] \in \mathbb{R}^{K\times L}$. As it can be seen, the coefficients are real-valued variables that correspond to the significance of each basis in the linear combination. We

can further decompose this into,

$$\mathbf{w}^{(i)} = \mathbf{z}^{(i)} \odot \mathbf{s}^{(i)}, \tag{3.29}$$

where $\mathbf{z}^{(i)} \in \{0,1\}^L$ is Bernoulli distributed and $\mathbf{s}^{(i)} \in \mathbb{R}^L$ is normally distributed. The collection of $\mathbf{z}^{(i)} \forall i$ in the binary matrix, $\mathbf{Z} = \left[\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, ..., \mathbf{z}^{(T)}\right]^T \in \{0,1\}^{T \times L}$ provides the modelling requirements for Figure 3.5(b). We will develop on this modelling assumption in chapter 5 with the introduction of the Beta Process Factor Analysis (BPFA) (Paisley and Carin, 2009; Zhou et al., 2012).

3.5 Model parameters and Bayesian regression

So far the discussion was given for models that have latent variables, that is for each new data point a respective latent variable is created. There are other types of Bayesian models which do not follow this principle. These models have fixed parameters before observing any data and remain with the same number of parameters throughout the training process, by adjusting the parameters' values only and not their size. We will describe one such Bayesian regression model with fixed number of parameters, the model coefficients. This will be useful in understanding the Relevance Vector Machine (RVM) that we will introduce in chapter 4.

Consider a collection of training data $\mathbf{K} = \{\mathbf{k}^{(i)}\}_{i=1}^N$ and $\mathbf{t} = \{t^{(i)}\}_{i=1}^N$, with each $\mathbf{k}^{(i)} \in \mathbb{R}^c$ having a corresponding target value, $t^{(i)} \in \mathbb{R}$. A regression problem is the prediction of $t^{(*)}$ from an unseen $\mathbf{k}^{(*)}$. To be able to make predictions, a model has to be constructed that effectively describes the training data. A popular first step is to create a linear model given by

$$t^{(i)} = \mathbf{w}^T \mathbf{k}^{(i)} + \epsilon^{(i)}, \tag{3.30}$$

where $\mathbf{w} \in \mathbb{R}^c$ is the vector of coefficients of the linear combination of the input data and $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$ is independent and identically normally distributed (i.i.d.) additive noise to accommodate any modelling errors. This translates to a likelihood function,

$$p(t^{(i)}|\mathbf{w}, \mathbf{k}^{(i)}) = \mathcal{N}(t^{(i)}; \mathbf{w}^T \mathbf{k}^{(i)}, \sigma^2) \quad = \frac{1}{\sqrt{2\pi\sigma^2}} \mathrm{e}^{-\frac{1}{2\sigma^2}(t^{(i)} - \mathbf{w}^T \mathbf{k}^{(i)})^2}. \tag{3.31}$$

Using the i.i.d. assumption,

$$p(\mathbf{t}|\mathbf{w}, \mathbf{K}) = \prod_{i=1}^{N} p(t^{(i)}|\mathbf{w}, \mathbf{k}^{(i)}), \tag{3.32}$$

where $\mathbf{t} \in \mathbb{R}^N$ and $\mathbf{K} \in \mathbb{R}^{c \times N}$. To use the Bayes' rule, we need to specify a prior distribution on the model parameters, $\mathbf{w}$. This choice reflects any prior beliefs about the model and it plays a crucial role in the calculation of the posterior distribution. In order to make calculations easier, a conjugate normal prior is defined as

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \alpha^{-1}\mathbf{I}) \quad = \sqrt{\frac{(\alpha)^L}{(2\pi)^L}} \mathrm{e}^{-\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}}. \tag{3.33}$$

where α is the precision (inverse of variance) of the normal distribution. Using Bayes' rule, the posterior distribution is given by,

$$p(\mathbf{w}|\mathbf{t}, \mathbf{K}) = \frac{p(\mathbf{t}|\mathbf{w}, \mathbf{K})p(\mathbf{w})}{\int_{\mathbf{w}} p(\mathbf{t}|\mathbf{w}, \mathbf{K})p(\mathbf{w})d\mathbf{w}}. \tag{3.34}$$

The above expression can be re-written as,

$$p(\mathbf{w}|\mathbf{t}, \mathbf{K}) \propto p(\mathbf{t}|\mathbf{w}, \mathbf{K})p(\mathbf{w}), \tag{3.35}$$

where $\propto$ means proportional since we dropped the normalising constant in the denominator of equation 3.34. Using the expressions for the likelihood function and prior probability distributions,

$$p(\mathbf{w}|\mathbf{t}, \mathbf{K}) \propto \left[\prod_{i=1}^{N} \mathrm{e}^{-\frac{1}{2\sigma^2}(t^{(i)} - \mathbf{w}^T\mathbf{k}^{(i)})^2}\right] \left[\mathrm{e}^{-\frac{\alpha}{2}\mathbf{w}^T\mathbf{w}}\right] \propto \mathrm{e}^{-\frac{\alpha}{2}\mathbf{w}^T\mathbf{w} - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(t^{(i)} - \mathbf{w}^T\mathbf{k}^{(i)})^2}, \tag{3.36}$$

where the normalising constants can be dropped. We re-organise this expression by expanding the square and then separating the dependent and independent terms with respect to $\mathbf{w}$. This results in

$$p(\mathbf{w}|\mathbf{t}, \mathbf{K}) \propto \exp\left\{-\frac{1}{2}\left[\mathbf{w}^T(\alpha\boldsymbol{I} + \frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{k}^{(i)}(\mathbf{k}^{(i)})^T)\mathbf{w} - 2\mathbf{w}^T(\frac{1}{\sigma^2}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)}) + \frac{1}{\sigma^2}\sum_{i=1}^{N}(t^{(i)})^2\right]\right\}$$

$$p(\mathbf{w}|\mathbf{t}, \mathbf{K}) \propto \exp\left\{-\frac{1}{2}\left[\mathbf{w}^T(\alpha\boldsymbol{I} + \frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{k}^{(i)}(\mathbf{k}^{(i)})^T)\mathbf{w} - 2\mathbf{w}^T(\frac{1}{\sigma^2}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)})\right]\right\}$$
$$* \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(t^{(i)})^2\right\}$$

We can then drop the last exponential term since it is independent of $\mathbf{w}$, and thus constant. It is absorbed in the proportionality resulting in,

$$p(\mathbf{w}|\mathbf{t},\mathbf{K}) \quad \propto \quad \exp\left\{-\frac{1}{2}\left[\mathbf{w}^T(\alpha\boldsymbol{I}+\frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{k}^{(i)}(\mathbf{k}^{(i)})^T)\mathbf{w}-2\mathbf{w}^T(\frac{1}{\sigma^2}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)})\right]\right\}$$

Then, the goal is to complete the square in the exponent again. To do this, we need to multiply by a term that is independent of $\mathbf{w}$. Thus,

$$\begin{aligned} p(\mathbf{w}|\mathbf{t},\mathbf{K}) \propto \exp\left\{-\frac{1}{2}\left[\mathbf{w}^T(\alpha\boldsymbol{I}+\frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{k}^{(i)}(\mathbf{k}^{(i)})^T)\mathbf{w}-2\mathbf{w}^T(\frac{1}{\sigma^2}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)})\right]\right\} \\ \exp\left\{-\frac{1}{2}\left[(\frac{1}{\sigma^2}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)})^T(\alpha\boldsymbol{I}+\frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{k}^{(i)}(\mathbf{k}^{(i)})^T)^{-1}(\frac{1}{\sigma^2}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)})\right]\right\} \end{aligned} \tag{3.37}$$

By using,

$$\boldsymbol{\Sigma} = (\alpha\boldsymbol{I}+\frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{k}^{(i)}(\mathbf{k}^{(i)})^T)^{-1} \tag{3.38}$$

and

$$\boldsymbol{\mu} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}\sum_{i=1}^{N}t^{(i)}\mathbf{k}^{(i)} \tag{3.39}$$

in equation (3.37), re-arranging the terms, it can be written as

$$p(\mathbf{w}|\mathbf{t},\mathbf{K}) \propto \exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\} \tag{3.40}$$

The goal is to obtain an expression for the posterior distribution with the exact formula given by

$$p(\mathbf{w}|\mathbf{t},\mathbf{K}) = \frac{\exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\}}{\int_{\mathbf{w}}\exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\}d\mathbf{w}} \tag{3.41}$$

Solving the integral in the denominator is not trivial, however, using the definition of a multivariate normal distribution, the normalising constant is given by $\frac{1}{\sqrt{(2\pi)^L|\boldsymbol{\Sigma}|}}$ where $|.|$ is the matrix's determinant. Therefore, the posterior distribution is given by a multivariate normal distribution

$$p(\mathbf{w}|\mathbf{t},\mathbf{K}) = \frac{\exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\}}{\sqrt{(2\pi)^L|\boldsymbol{\Sigma}|}} \tag{3.42}$$

with covariance, $\boldsymbol{\Sigma}$ and mean, $\boldsymbol{\mu}$ as defined in equations 3.38 and 3.39 respectively.

Note that the posterior distribution obtained is in the same family of distributions as the prior distribution chosen in equation 3.33. This shows that the normal prior distribution is a conjugate prior to the normal likelihood function which leads to a posterior distribution from the same family but with different parameters. Having a conjugate prior is very useful since it is only necessary to update the posterior distribution parameters by using the observed data.

Obtaining the posterior distribution provides useful information about the relationship between different model variables. However, the main motivation in learning a model is to utilise it for making predictions for new data points. In the case of the Bayesian regression model, the aim is to predict a new value $t^{(*)}$ from $\mathbf{k}^{(*)}$. Using the posterior distribution of the model parameters, it is possible to construct a predictive distribution given by

$$p(t^{(*)}|\mathbf{k}^{(*)}, \mathbf{t}, \mathbf{K}) = \int_{\mathbf{w}} p(t^{(*)}|\mathbf{k}^{(*)}, \mathbf{w}) p(\mathbf{w}, |\mathbf{t}, \mathbf{K}) d\mathbf{w}. \tag{3.43}$$

Looking closer at equation 3.43, the model parameters, $\mathbf{w}$, are not known. Nevertheless, by marginalising over all their possible configurations, it is possible to obtain a weighted solution. The posterior distribution is used as a weight providing the belief of how probable a certain configuration is.

For the Bayesian regression problem that was discussed, it is possible to solve this integral analytically as opposed to other methods that we discussed previously that required sampling using MCMC. Here we can directly write the likelihood and posterior distribution in the integral giving

$$p(t^{(*)}|\mathbf{k}^{(*)}, \mathbf{t}, \mathbf{K}) = \int_{\mathbf{w}} \frac{\exp\left\{-\frac{1}{2\sigma^2}(t^{(*)} - \mathbf{w}^T\mathbf{k}^{(*)})^2\right\}}{\sqrt{2\pi\sigma^2}} \frac{\exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w} - \boldsymbol{\mu})\right\}}{\sqrt{(2\pi)^L|\boldsymbol{\Sigma}|}} d\mathbf{w}. \tag{3.44}$$

This can be re-written as

$$\begin{aligned} p(t^{(*)}|\mathbf{k}^{(*)}, \mathbf{t}, \mathbf{K}) = {} & \frac{1}{\sqrt{2\pi\sigma^2}} \frac{1}{\sqrt{(2\pi)^L|\boldsymbol{\Sigma}|}} \exp\left\{-\frac{1}{2\sigma^2}(t^{(*)})^2 - \frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right\} \\ & * \int_{\mathbf{w}} \exp\left\{-\frac{1}{2}\left(\mathbf{w}^T\left(\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2} + \boldsymbol{\Sigma}^{-1}\right)\mathbf{w} - 2\mathbf{w}^T\left(\frac{\mathbf{k}^{(*)}t^{(*)}}{\sigma^2} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right)\right)\right\} d\mathbf{w} \end{aligned} \tag{3.45}$$

where the squares in the exponents were expanded and the terms independent from $\mathbf{w}$ are separated outside the integral. This is done in order to construct a normal distribution

inside the integral that would sum to 1. To do this, we multiply and divide by

$$\frac{1}{\sqrt{(2\pi)^L}}|\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2}+\mathbf{\Sigma}^{-1}|^{\frac{1}{2}}\exp\left\{-\frac{1}{2}(\frac{t^{(*)}\mathbf{k}^{(*)}}{\sigma^2}+\mathbf{\Sigma}^{-1}\boldsymbol{\mu})^T(\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2}+\mathbf{\Sigma}^{-1})^{-1}(\frac{t^{(*)}\mathbf{k}^{(*)}}{\sigma^2}+\mathbf{\Sigma}^{-1}\boldsymbol{\mu})\right\} \tag{3.46}$$

By multiplying this term inside the integral, it is possible to complete the square, resulting in a normal distribution. Thus, the integral is 1 and the integral term in the numerator vanishes. The predictive distribution becomes, $p(t^{(*)}|\mathbf{k}^{(*)},\mathbf{t},\mathbf{K}) =$

$$\frac{\frac{1}{\sqrt{2\pi\sigma^2}}\frac{1}{\sqrt{2\pi^L|\mathbf{\Sigma}|}}\exp\left\{-\frac{1}{2\sigma^2}(t^{(*)})^2-\frac{1}{2}\boldsymbol{\mu}^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}\right\}}{\frac{1}{\sqrt{(2\pi)^L}}|\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2}+\mathbf{\Sigma}^{-1}|^{\frac{1}{2}}\exp\left\{-\frac{1}{2}(\frac{t^{(*)}\mathbf{k}^{(*)}}{\sigma^2}+\mathbf{\Sigma}^{-1}\boldsymbol{\mu})^T(\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2}+\mathbf{\Sigma}^{-1})^{-1}(\frac{t^{(*)}\mathbf{k}^{(*)}}{\sigma^2}+\mathbf{\Sigma}^{-1}\boldsymbol{\mu})\right\}} \tag{3.47}$$

Using the matrix determinant lemma,

$$|\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2}+\mathbf{\Sigma}^{-1}| = |\mathbf{\Sigma}|^{-1}(1+\frac{(\mathbf{k}^{(*)})^T\mathbf{\Sigma}\mathbf{k}^{(*)})}{\sigma^2}) \tag{3.48}$$

and the Woodbury identity,

$$(\frac{\mathbf{k}^{(*)}(\mathbf{k}^{(*)})^T}{\sigma^2}+\mathbf{\Sigma}^{-1})^{-1} = \mathbf{\Sigma}-\mathbf{\Sigma}\mathbf{k}^{(*)}(\sigma^2+(\mathbf{k}^{(*)})^T\mathbf{\Sigma}\mathbf{k}^{(*)})^{-1}(\mathbf{k}^{(*)})^T\mathbf{\Sigma}, \tag{3.49}$$

and replacing and cancelling terms, combining the ratio of exponentials results in

$$p(t^{(*)}|\mathbf{k}^{(*)},\mathbf{t},\mathbf{K}) = \frac{1}{\sqrt{2\pi(\sigma^2+(\mathbf{k}^{(*)})^T\mathbf{\Sigma}\mathbf{k}^{(*)})}}\exp\left\{-\frac{1}{2(\sigma^2+(\mathbf{k}^{(*)})^T\mathbf{\Sigma}\mathbf{k}^{(*)})}(t^{(*)}-(\mathbf{k}^{(*)})^T\boldsymbol{\mu})^2\right\} \tag{3.50}$$

Therefore, the predictive distribution is again a normal distribution, with

$$\text{Predictive mean} = (\mathbf{k}^{(*)})^T\boldsymbol{\mu} \tag{3.51}$$

and

$$\text{Predictive variance} = \sigma^2+(\mathbf{k}^{(*)})^T\mathbf{\Sigma}\mathbf{k}^{(*)}. \tag{3.52}$$

We have shown that it is possible to obtain a predictive distribution for a new data point, $t^{(*)}$, by using the posterior distribution of the model parameters, $\mathbf{w}$ and the likelihood function analytically. We will use this in chapters 4 and 7 to introduce the Relevance Vector Machine (RVM). The RVM will be used in the context of seismic acquisition for prediction and uncertainty quantification.

Chapter 4

The Relevance Vector Machine for Seismic Compressive Sensing

Bayesian statistics and machine learning provide a framework to create probabilistic data-driven models using probability distributions that capture assumptions about the data. One of the necessary assumptions for Compressive Sensing (CS) to work is that the seismic signals are sparse in some transform. In this chapter, we will use a sparse Bayesian regression model, called the Relevance Vector Machine (RVM) (Tipping, 2001). It is similar to the one described in section 3.5 but with some enhancements that we will describe. We start the discussion with a description of the RVM and its fast version. Then, we provide a description of an extension of the RVM along with a description of evaluation criteria, domains of processing and parameter tuning. Comparisons with other algorithms are provided both in reconstruction accuracy and in computational time along with representative examples.

4.1 The Relevance Vector Machine

The probabilistic data-driven model described in section 3.5 assumed a linear relationship between inputs and outputs. This is not realistic in many real world applications and thus, a dictionary of basis functions is used to transform the input space. The RVM is one such model that uses basis functions which are assumed pre-defined, fixed and non-linear. The problem is still linear in the model parameters, $\mathbf{w}$, but now in a transformed input space. The model becomes (Tipping, 2001),

$$t^{(i)} = \sum_{l=1}^{L} w_l \boldsymbol{\phi}_l(\mathbf{k}^{(i)}) + \epsilon^{(i)} = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{k}^{(i)}) + \epsilon^{(i)}, \tag{4.1}$$

where $\mathbf{w} \in \mathbb{R}^L$ are the coefficients of the linear combination of the transformed input data, $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$ is independent and identically distributed (i.i.d.) additive noise and $\boldsymbol{\phi}(\mathbf{k}^{(i)}) = \left[\boldsymbol{\phi}_1(\mathbf{k}^{(i)}), \boldsymbol{\phi}_2(\mathbf{k}^{(i)}), ..., \boldsymbol{\phi}_L(\mathbf{k}^{(i)})\right]^T \in \mathbb{R}^L$ with each entry being a certain basis function applied to the particular data point, i. The corresponding likelihood function is given by

$$p(t^{(i)}|\mathbf{w}, \boldsymbol{\phi}(\mathbf{k}^{(i)}), \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(t^{(i)} - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{k}^{(i)}))^2\right\} \tag{4.2}$$

Using the i.i.d. assumption,

$$p(\mathbf{t}|\mathbf{w}, \boldsymbol{\Phi}, \sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(t^{(i)} - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{k}^{(i)}))^2\right\} \tag{4.3}$$

$$= \frac{1}{(\sqrt{2\pi\sigma^2})^N} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}\|^2\right\}, \tag{4.4}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{N\times L}$. It is given by $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_L]$ where each $\boldsymbol{\phi}_l \in \mathbb{R}^N$ is the l-th basis function evaluated at N available receivers.

For Compressive Sensing applications, there is a required assumption that the acquired signal lives in a sparse domain (or it can be transformed in a sparse domain). Thus, a prior probability distribution on the model parameters, $\mathbf{w}$, that promotes sparsity is required. As before, a normal prior distribution is preferred, which is conjugate to the likelihood function. In this case, each coefficient, w_l, is associated with a different variance which is controlled by the precision, α_l. Thus, the prior distribution is given by

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{l=1}^{L} \mathcal{N}(w_l; 0, \alpha_l^{-1}) = \prod_{l=1}^{L} \sqrt{\frac{\alpha_l}{2\pi}} e^{\frac{-\alpha_l}{2}w_l^2}, \tag{4.5}$$

which is a product of zero mean normal distributions with each distribution having a precision, α_l. Note that, the prior distribution is now conditioned on $\boldsymbol{\alpha} \in \mathbb{R}^L$ given by $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_L]^T$. These are called hyperparameters and they are governed by a hyperprior distribution. Since these parameters scale the normal distributions according to their value, suitable priors are the Gamma distributions given by

$$p(\boldsymbol{\alpha}) = \prod_{l=1}^{L} \text{Gamma}(\alpha_l; c, d) = \prod_{l=1}^{L} \frac{1}{\Gamma(c)} d^c \alpha_l^{c-1} e^{-d\alpha_l} \tag{4.6}$$

where $\Gamma(c) = \int_0^\infty t^{c-1}e^{-t}dt$ is the Gamma function. The noise precision, σ^{-2}, is also modelled by a Gamma distribution,

$$p(\sigma^{-2}) = \text{Gamma}(\sigma^{-2}; e, f). \tag{4.7}$$

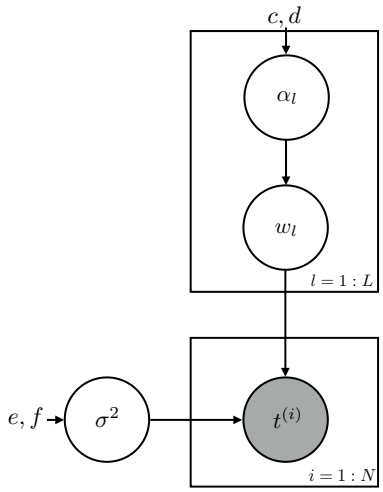


Fig. 4.1 Graphical model of the Relevance Vector Machine (RVM) model illustrating the hierarchical prior on the model parameters.

Typically, the Gamma distributions are chosen to be non-informative and the parameters that control them are set to very small values such as $c = d = e = f = 10^{-6}$ (Tipping, 2001). This results in a hierarchical prior formulation which provides flexibility in the coefficients. It allows some probability mass to potentially concentrate on a few coefficients and others to be zero or close to zero resulting in the desired property of sparsity. That is, in the inference stage some coefficients are deemed *relevant* by the inference algorithm, and some tend to zero, hence the name of the Relevance Vector Machine (RVM). A summary of the RVM model is given in the graphical model of Figure 4.1. From the graphical model and the Bayes' rule, we can get the posterior distribution of the unknown model parameters by,

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{t}, \boldsymbol{\Phi}) = \frac{p(\mathbf{t}|\mathbf{w}, \boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2) p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)}{p(\mathbf{t})} \tag{4.8}$$

$$= \frac{p(\mathbf{t}|\mathbf{w}, \boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2) p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)}{\int p(\mathbf{t}|\mathbf{w}, \boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2) p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2) d\mathbf{w} d\boldsymbol{\alpha} d\sigma^2}. \tag{4.9}$$

In this problem, the denominator is an integral over three variables and in high dimensional spaces and cannot be solved directly analytically as before. In fact, not being able to directly solve the normalising constant of Bayes' rule is typical in Bayesian modelling problems and a lot of research has been done towards its approximation. MCMC methods that we described in section 3.2 are one such attempt.

For the RVM, a different approach is followed towards the approximation of the posterior distribution over the unknown parameters. It is split into two parts such as,

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t}, \boldsymbol{\Phi}) = p(\mathbf{w}|\mathbf{t}, \boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2)p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}, \boldsymbol{\Phi}), \tag{4.10}$$

where the product rule is used. The posterior distribution of the coefficients, $\mathbf{w}$, can be further decomposed in,

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\boldsymbol{\Phi}, \mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2)} \tag{4.11}$$

$$= \frac{p(\mathbf{t}|\boldsymbol{\Phi}, \mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{\int p(\mathbf{t}|\boldsymbol{\Phi}, \mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}}. \tag{4.12}$$

This is very similar to the Bayesian regression problem in equation 3.34 with two main differences. Here, there is one separate precision, α_l, for each coefficient and also basis functions incorporated in a basis matrix, $\boldsymbol{\Phi}$ are used. Therefore, the posterior distribution is a normal distribution given by,

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\Phi}, \boldsymbol{a}, \sigma^2) = \frac{1}{\sqrt{(2\pi)^L|\boldsymbol{\Sigma}|}} \exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w} - \boldsymbol{\mu})\right\} \tag{4.13}$$

where L is the dimension of the coefficients. The covariance, $\boldsymbol{\Sigma}$, and mean, $\boldsymbol{\mu}$ are now given by

$$\boldsymbol{\Sigma} = (\sigma^{-2}\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \boldsymbol{A})^{-1}, \tag{4.14}$$

$$\boldsymbol{\mu} = \sigma^{-2}\boldsymbol{\Sigma}\boldsymbol{\Phi}^T\mathbf{t}. \tag{4.15}$$

where $\boldsymbol{A}$ is the diagonal matrix, $\text{diag}(\alpha_1, \alpha_2, ..., \alpha_L)$. Inference then involves finding the optimum configuration of $\boldsymbol{\alpha}$ and σ^2 that maximise $p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}, \boldsymbol{\Phi})$. An approximation is required as it is not possible to obtain the full posterior over the unknowns. This approximation is a delta function at the mode of the probability distribution and is based on the fact that the most probable values $\boldsymbol{\alpha}$ and σ^2 are close to the ones sampled from the full posterior. Therefore, it is necessary to maximise,

$$p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}, \boldsymbol{\Phi}) \propto p(\mathbf{t}|\boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2)p(\boldsymbol{\alpha})p(\sigma^2). \tag{4.16}$$

It is only necessary to maximise the likelihood term as the two hyper-priors are set as non-informative. This is given by (Tipping, 2001),

$$p(\mathbf{t}|\mathbf{\Phi}, \boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{t}|\mathbf{\Phi}, \mathbf{w}, \sigma^2) p(\mathbf{w}|\boldsymbol{\alpha}) d\mathbf{w} \tag{4.17}$$

$$= \frac{1}{\sqrt{(2\pi)^N}} |\sigma^2 \boldsymbol{I}_N + \mathbf{\Phi} \boldsymbol{A}^{-1} \mathbf{\Phi}^T|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\mathbf{t}^T (\sigma^2 \boldsymbol{I}_N + \mathbf{\Phi} \boldsymbol{A}^{-1} \mathbf{\Phi}^T)^{-1} \mathbf{t}\right\}. \tag{4.18}$$

Instead of maximising the above expression, its natural logarithm is used which is mathematically more convenient. So, the log-likelihood, $\mathcal{L}$, is given by (Tipping, 2001)

$$\begin{aligned}\mathcal{L} &= \ln\left\{\frac{1}{\sqrt{(2\pi)^N}} |\sigma^2 \boldsymbol{I}_N + \mathbf{\Phi} \boldsymbol{A}^{-1} \mathbf{\Phi}^T|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\mathbf{t}^T (\sigma^2 \boldsymbol{I}_N + \mathbf{\Phi} \boldsymbol{A}^{-1} \mathbf{\Phi}^T)^{-1} \mathbf{t}\right\}\right\} \\ &= -\frac{1}{2}\ln|\sigma^2 \boldsymbol{I}_N + \mathbf{\Phi} \boldsymbol{A}^{-1} \mathbf{\Phi}^T| - \frac{1}{2}\left[\mathbf{t}^T (\sigma^2 \boldsymbol{I}_N + \mathbf{\Phi} \boldsymbol{A}^{-1} \mathbf{\Phi}^T)^{-1} \mathbf{t}\right],\end{aligned} \tag{4.19}$$

where the constants that are independent of $\boldsymbol{\alpha}$ and σ^2 are dropped.

The original algorithm of the RVM (Tipping, 2001) uses an iterative procedure that obtains the parameters that maximise the expression. At some point of the re-estimations, the majority of elements in $\boldsymbol{\alpha}$ tend to infinity, which means that the coefficients associated with the large precisions tend to zero. This way, the desired sparsity effect is achieved and the corresponding basis function is removed from the model. In this thesis a fast version of the RVM will be used and is described in the next section.

4.2 Fast Relevance Vector Machine

In the previous section, we introduced the Relevance Vector Machine (RVM) as a Bayesian Compressive Sensing algorithm. The procedure for obtaining the model parameters requires the estimation of data statistics to obtain the covariance and mean. However, this requires the inversion of the covariance matrix as seen in equation 4.14, at every iteration, which makes the entire procedure computationally expensive. Tipping and Faul (2003) proposed an algorithm to obtain the model parameters faster. This is based on the analysis of the marginal likelihood of the data which was presented by Faul and Tipping (2001). We re-write the log-likelihood of the model which was presented in equation 4.19 and replace some terms to facilitate the discussion. That is,

$$\mathcal{L} = -\frac{1}{2}\ln|\mathbf{C}| - \frac{1}{2}\mathbf{t}^T \mathbf{C}^{-1} \mathbf{t} \tag{4.20}$$

where $\mathbf{C} = \sigma^2\boldsymbol{I} + \boldsymbol{\Phi}\boldsymbol{A}^{-1}\boldsymbol{\Phi}^T$. We can split $\mathbf{C}$ into hyper-parameter dependencies,

$$\mathbf{C} = \sigma^2\mathbf{I} + \sum_{l\neq j}(\alpha_l)^{-1}\boldsymbol{\phi}_l(\boldsymbol{\phi}_l)^T + (\alpha_j)^{-1}\boldsymbol{\phi}_j(\boldsymbol{\phi}_j)^T, \tag{4.21}$$

where $\boldsymbol{\phi}_j$ is the j-th basis function applied to all available data points. Using the above, another definition is given by

$$\mathbf{C}_{-j} = \sigma^2\mathbf{I} + \sum_{l\neq j}(\alpha_l)^{-1}\boldsymbol{\phi}_l(\boldsymbol{\phi}_l)^T. \tag{4.22}$$

Using the two identities defined earlier in equations 3.48 and 3.49, we can write the expressions of interest that appear in the log-likelihood as

$$|\mathbf{C}| = |\mathbf{C}_{-j}||1 + (\alpha_j)^{-1}(\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j| \tag{4.23}$$

and

$$\mathbf{C}^{-1} = (\boldsymbol{C}_{-j})^{-1} - \frac{(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j(\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}}{\alpha_j + (\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j}. \tag{4.24}$$

We can then replace these expressions in the log-likelihood to obtain,

$$\begin{aligned}
\mathcal{L} &= -\frac{1}{2}\ln\left[|\mathbf{C}_{-j}||1 + (\alpha_j)^{-1}(\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j|\right] - \frac{1}{2}\mathbf{t}^T\left[(\boldsymbol{C}_{-j})^{-1} - \frac{(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j(\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}}{\alpha_j + (\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j}\right]\mathbf{t} \\
&= -\frac{1}{2}\ln|\mathbf{C}_{-j}| + \frac{1}{2}\ln\alpha_j - \frac{1}{2}\ln|\alpha_j + (\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j| - \frac{1}{2}\mathbf{t}^T(\boldsymbol{C}_{-j})^{-1}\mathbf{t} + \frac{1}{2}\frac{((\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\mathbf{t})^2}{\alpha_j + (\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j} \\
&= \mathcal{L}_{-j} + \frac{1}{2}\left[\ln\alpha_j - \ln(\alpha_j + s_j) + \frac{(q_j)^2}{\alpha_j + s_j}\right]
\end{aligned}$$

where $\mathcal{L}_{-j}$ contains all the variables of the log-likelihood without the effect of α_j. The rest of the expression only contains variables depending on α_j which can be used to obtain the derivatives that maximise the expression. To simplify the expression, we defined two further variables as it was done by Tipping and Faul (2003) given by

$$s_j = (\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\boldsymbol{\phi}_j \tag{4.25}$$

and

$$q_j = (\boldsymbol{\phi}_j)^T(\boldsymbol{C}_{-j})^{-1}\mathbf{t}. \tag{4.26}$$

s_j is called the sparsity factor and gives a measure of how much the basis function, $\boldsymbol{\phi}_j$, overlaps with the basis functions already in the model. This serves to decrease $\mathcal{L}$ by

adding to the normalising constant. q_j on the other hand is called the quality factor and gives a measure of how well $\boldsymbol{\phi}_j$ increases $\mathcal{L}$ by helping to explain the data.

Using the above expressions and the analysis by Faul and Tipping (2001), it can be shown that the log-likelihood has a maximum when

$$\alpha_j = \begin{cases} \frac{s_j^2}{q_j^2 - s_j} & \text{if } q_j^2 > s_j \\ \infty & \text{if } q_j^2 \leq s_j. \end{cases} \tag{4.27}$$

As we mentioned earlier, it is desirable that the majority of elements in $\boldsymbol{\alpha}$ tend to infinity which means that the majority of coefficients tend to zero. This ensures the sparsity assumption of the coefficients and the corresponding basis functions are also removed from the model. Thus, if $\boldsymbol{\phi}_j$ is already in the model and $q_j^2 \leq s_j$ then we could delete $\boldsymbol{\phi}_j$ from the model. At the same time, if $\boldsymbol{\phi}_j$ is not in the model and $q_j^2 > s_j$, then we could add this in the model. This was used by Tipping and Faul (2003) to create a sequential algorithm for sparse Bayesian estimation. A summary of the algorithm can be seen in Algorithm 3.

There are different possibilities for this algorithm. First, the initialisation of the noise variance, σ^2, should be such that the noise is not over-estimated. An empirical study for the SEAM-II dataset is provided in section 4.4.2. The choice of the first basis function is done by finding the largest normalised projection on to the available data. That is, the algorithm searches for the largest $\frac{\|\boldsymbol{\phi}_l^T \mathbf{t}\|^2}{\|\boldsymbol{\phi}_l\|^2}$ for all l and the basis function, $\boldsymbol{\phi}_l$ with the largest projection, is chosen to initialise the model. This is then used to calculate the first α_l as,

$$\alpha_l = \frac{\|\boldsymbol{\phi}_l\|^2}{\|\boldsymbol{\phi}_l^T \mathbf{t}\|^2 / \|\boldsymbol{\phi}_l\|^2 - \sigma^2}. \tag{4.28}$$

Algorithm 3 Fast RVM

1: **procedure** FastRVM
2: Initialise $\sigma^2, \boldsymbol{\phi}_l$ and α_l [Details in text]
3: Compute $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}$ and $s_l, q_l \; \forall l$
4: Choose a basis function $\boldsymbol{\phi}_l$ from the dictionary [Details in text]
5: Calculate $\theta_l = q_l^2 - s_l$
6: **if** $(\theta_l > 0)$ AND $(\alpha_l < \infty)$ **then** re-estimate α_l
7: **if** $(\theta_l > 0)$ AND $(\alpha_l = \infty)$ **then** add α_l
8: **if** $(\theta_l \leq 0)$ AND $(\alpha_l < \infty)$ **then** delete α_l
9: If not converged, go back to 4.
10: **end**

In subsequent iterations, the algorithm needs to choose a basis function from the dictionary and decide what to do with it. This can be done at random or from a predefined list. In order to find the greatest increase in the log-likelihood per iteration, a formula is given in the Appendix of Tipping and Faul (2003) along with update formulas for the various options that the algorithm can take for either step 6, 7 or 8. The last consideration is the convergence criterion. If there is no significant difference in the $\log \boldsymbol{\alpha}$ for all basis functions then the algorithm terminates. A suitable threshold is proposed as 10^{-6} by Tipping and Faul (2003).

After the algorithm terminates, it returns the statistics of the model which are $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}$. These are used in the predictive distribution given by

$$p(t^{(*)}|\mathbf{k}^{(*)}, \mathbf{t}, \boldsymbol{\Phi}, \boldsymbol{\alpha}, \sigma^2) = \int p(t^{(*)}|\mathbf{k}^{(*)}, \boldsymbol{\Phi}, \mathbf{w}, \sigma^2) p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) d\mathbf{w}. \tag{4.29}$$

It is similar to the Bayesian regression problem in section 3.5 and thus the predictive distribution is also a normal distribution $\mathcal{N}(t^{(*)}, \sigma_*^2)$ given by

$$t^{(*)} = m_* = \boldsymbol{\mu}^T \boldsymbol{\phi}(\mathbf{k}^{(*)}) \tag{4.30}$$

and the predictive variance

$$\sigma_*^2 = v_* = \sigma^2 + \boldsymbol{\phi}(\mathbf{k}^{(*)})^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{k}^{(*)}). \tag{4.31}$$

where $\boldsymbol{\phi}(\mathbf{k}^{(*)})$ is a vector of all basis functions, L, calculated at a missing receiver, $\mathbf{k}^{(*)}$. The desired behaviour is that we obtain small predictive variance when we are confident about the prediction and large predictive variance when we are uncertain. Nevertheless, this does not always occur and we will examine it further in chapter 7. We can also use it to create a cascade of RVMs as discussed in the next section.

4.3 Cascade of Relevance Vector Machines

Consider the scenario that a model uses basis functions with a finite support, for example Haar wavelets as described in section 2.4. If a data point is far from the centre or rather outside of the support, the basis function would evaluate to zero resulting in predictive variance, $\sigma_*^2 = \sigma^2$. This is not the desired behaviour. Furthermore, if none of the basis functions at location $\mathbf{k}^{(*)}$ are included in the model, then the predictive variance would also be $\sigma_*^2 = \sigma^2$ or exactly zero if we ignore the noise variance. This is counter-intuitive.

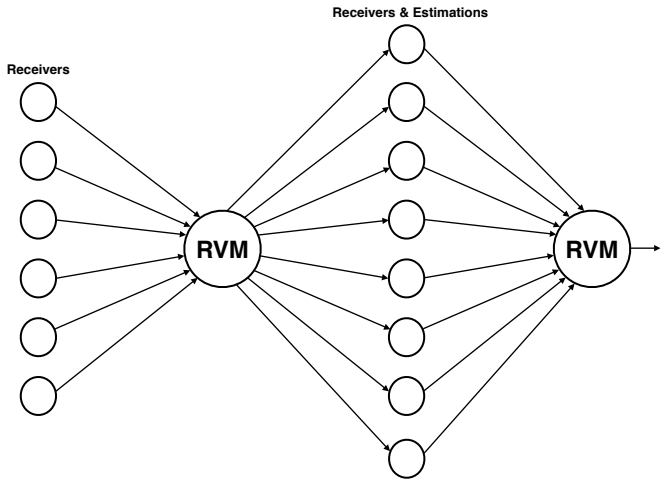


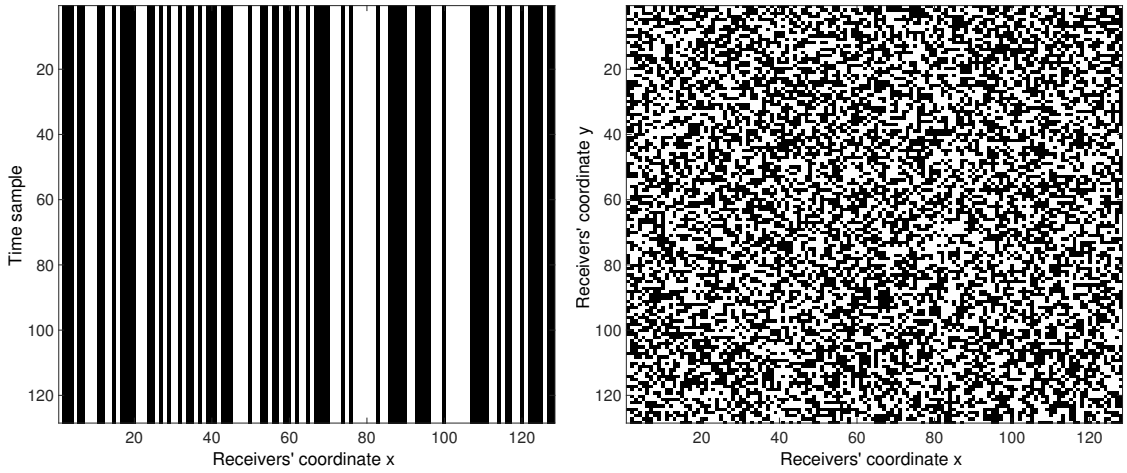
Fig. 4.2 Layers of Relevance Vector Machines that propagate the predictions for the missing receivers along with the original receivers' values to the next stage using the predictive variance as in equation 4.31. Each instance of the RVM is different from the other by the choice of the dictionary of basis functions.

If none of the basis functions which are non-zero at $\mathbf{k}^{(*)}$ are included in the model, the predictive variance should be large, however it is zero (ignoring the noise variance).

Let us think about the case when $\boldsymbol{\phi}(\mathbf{k}^{(*)}) = 0$. We can use this to inform the model not to trust the predictions in those regions. Using this information, we can create a deep model, using more layers of RVMs. Each RVM can utilise a different dictionary of basis functions and as input, it can use the output of the previous RVM by propagating only the predictions that are trustworthy. Figure 4.2 illustrates this network architecture. Predicted receivers' values are accepted and propagated through the network when the predictive variance is not zero. If the predictive variance is zero (ignoring the noise variance), they are still marked as missing and are propagated through for further estimation by another RVM with a different dictionary of basis functions with larger support. This cascade was used by Pilikos (2014) for image reconstruction. We will use this for seismic signal reconstruction. Before illustrating the results, we will first discuss about the evaluation criteria and the domains of operation.

4.4 Evaluation, domains and parameter tuning

The RVM is a sparse Bayesian model that is ideally suited for Compressive Sensing (CS). Due to the fact that it uses only a few non-zero coefficients as model parameters, it is able



(a) (b)

Fig. 4.3 Masks created by randomly traversing the 128×128 grid and drawing a number between 1 and 100. If the number is less or equal to the percentage, the receiver is kept and the index corresponds to one (white). If it is above, it is set to zero (black). (a) shows the mask for the x-t domain and (b) shows the mask for a time slice. Both examples set the percentage used as 50%.

to satisfy the sparsity assumption. In order to evaluate the performance of the RVM and the cascade, we extracted synthetic seismic data from the data set described in section 2.1, called SEAM-II provided by BP. The data were used in order to compare the algorithm with others in the literature mentioned in chapter 2. To evaluate the reconstruction accuracy of all algorithms, we will use the reconstruction quality, Q, defined by,

$$\mathrm{Q} = 10 \log_{10} \frac{\|\mathbf{x}\|_2^2}{\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2}, \tag{4.32}$$

where $\mathbf{x}$ is the original signal and $\hat{\mathbf{x}}$ is the reconstruction. This metric was used by Kazemi et al. (2016) for seismic data and it captures the error in the reconstruction by directly comparing the receivers' values of the predicted and original signal. We will also use the Frequency Wavenumber (FK) domain of the reconstructions to visualise if there is aliasing or incoherent noise in the frequency spectrum.

4.4.1 Domains and masks

There are two different domains that we can use for evaluation, namely the x-t domain (shot record) and the x-y domain (time slice). Missing receivers in each domain correspond to different patterns of removal as described by the domains in section 2.1. Figure 4.3(a)

shows the mask used for a section from the x-t domain. A receiver is removed by traversing all receivers and drawing a number from the uniform distribution between 1-100. If it is below or equal to the percentage used, the entire line is kept and marked as one otherwise marked as zero. Figure 4.3(b) shows the mask for a section of a time slice where each data point is a receiver and the entry in the mask is set to one if it is kept, otherwise to zero. We can see that within the mask for the x-t domain, there are numerous consecutive missing data points.

Thus, the training data available to the algorithms are more sparse with large gaps in between. In the case of time slices, the mask uses data points at random and with smaller gaps. The training data available for time slices provides a more balanced data set. We will use the RVM to reconstruct a section of a time slice with the same percentage of receivers used but using the two different masks from Figure 4.3. Figure 4.4 shows the original section from a time slice that we will use for the experiment.

Figure 4.5(a) shows the signal used by the RVM when using the mask for time slices for 50% of receivers used. Figure 4.5(b) shows the same signal but using the mask for the x-t domain for 50% of receivers used. The respective reconstructions are shown in Figure 4.5(c) and Figure 4.5(d). We can see that the reconstruction using the time slice mask from Figure 4.3(b) has better reconstruction quality, Q = 47.322 db as opposed to the reconstruction using the x-t domain mask from Figure 4.3(a) with Q = 12.524 db. In the subsequent experiments, we will be using time slices as the domain for reconstructing seismic signals for improved quality.

4.4.2 Tuning of the noise variance's initialisation

From Algorithm 3, describing the fast version of the RVM, we need to initialise model parameters such as the noise variance, σ^2, the first basis function in the model, $\boldsymbol{\phi}_l$ and the corresponding first precision, α_l for the corresponding model coefficient. The initialisation of the latter two has been described earlier. For the noise variance, an empirical study is performed but more specifically for the noise standard deviation, σ. Eleven different values for this are tested from $\sigma = 10^{-18}$ to $\sigma = 10^{-8}$ with a factor of ten increment. This range has been chosen since the seismic signals that we will use for evaluation have a standard deviation in the order of 10^{-10}. The original signal can be seen in Figure 4.6(a) and the same signal using only 50% of the receivers can be seen in Figure 4.6(b). We varied the noise standard deviation as mentioned and obtained various values for the reconstruction accuracy measured in Q as defined in equation 4.32. Figure 4.7 shows the reconstruction quality, Q, with different initialisations of the noise standard deviation.

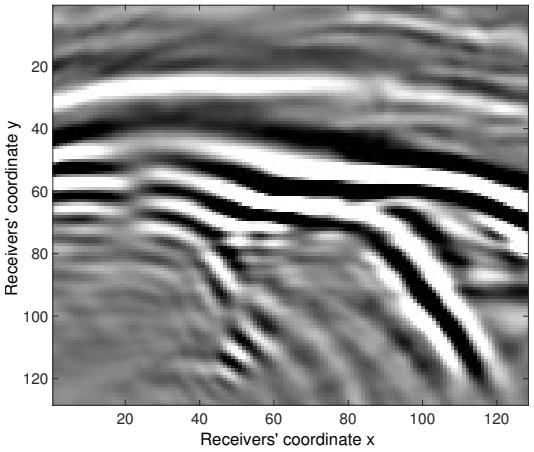


Fig. 4.4 Original section of 128 × 128 receivers from a time slice.

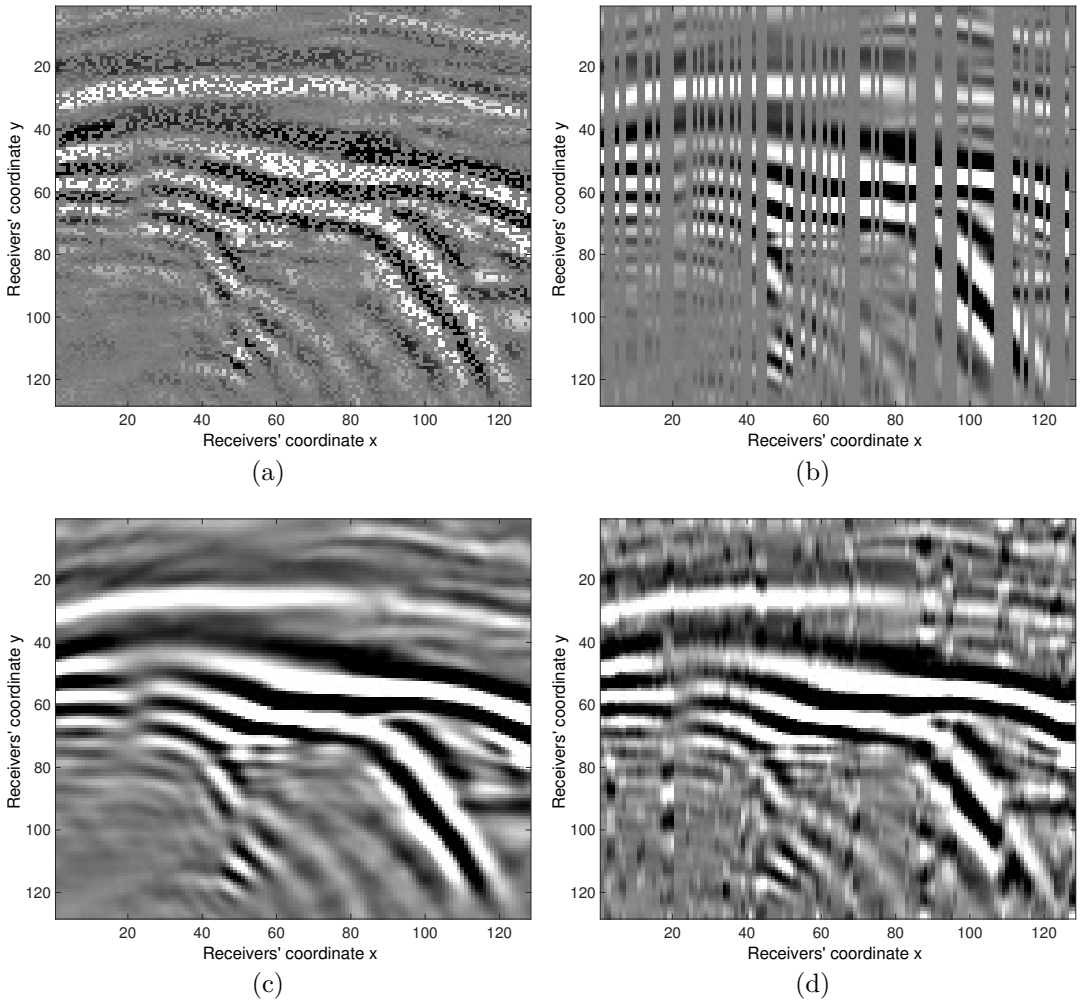


Fig. 4.5 A section using 50% randomly (a), using 50% of the lines (b), the reconstruction of (a) in (c) with Q = 47.322 db and the reconstruction of (b) in (d) with Q = 12.524 db.

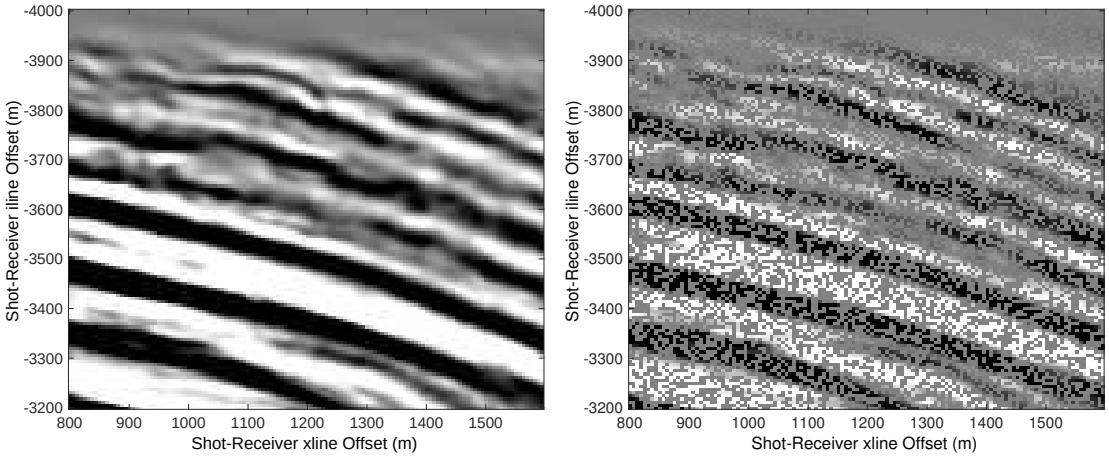


Fig. 4.6 Original section of 128×128 receivers extracted from a time slice (a) with the same signal using only 50% of receivers in (b).

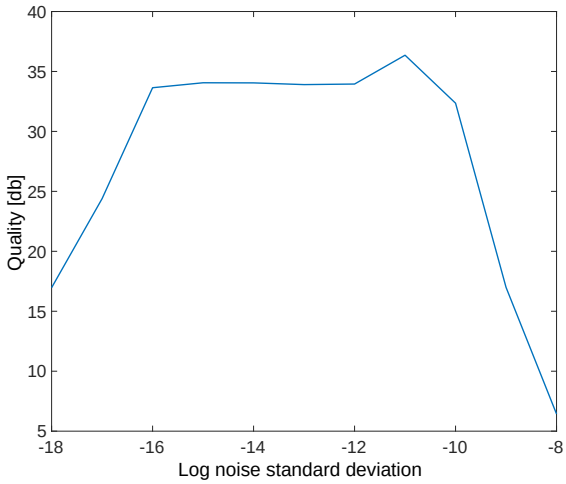


Fig. 4.7 Plot of the reconstruction quality, Q, against the log noise standard deviation. When the noise standard deviation is very small or very large the quality worsens.

The log noise standard deviation is plotted for visualisation convenience. It can be seen that when the noise standard deviation is very small or very large, the RVM behaves badly. If it is too small, it affects the calculation of the mean and the covariance of the model parameters since it scales them. If it is too large, the algorithm stops earlier than it should since it assumes that the remaining error is explained by the noise. There is a region where it obtains similar reconstruction accuracy when the noise standard deviation is from 10^{-16} to 10^{-12}. Nevertheless, the reconstruction accuracy peaks at 10^{-11} and this will be used throughout the experiments in this thesis.

4.4.3 Trade-off analysis between accuracy and time by tuning parameters

From the discussion so far, we have seen that there are tunable parameters that can change the configuration of the RVM. In addition, if we consider using the cascade of RVMs then we need to choose the number of layers, the patch size and the basis functions. Therefore, it is necessary to investigate the different parameters that can change in order to find the best configuration of the RVM. This will be useful when we compare the RVM against other algorithms in the literature.

First, we need to investigate whether the cascade of RVMs described in section 4.3 improves the reconstruction accuracy. Due to the sequential nature of the model, the more layers we use the longer the computational time. To test the accuracy of the network of RVMs, we used the two-dimensional multi-scale Haar Wavelet Transform described in chapter 2. By using the smallest support of 2×2, the finest details of the signal can be captured. The basis functions with support of 4×4 are used to capture larger regions and so on. In the following experiment, we used networks with one, two and three layers and for each we also varied the patch size between 8×8, 16×16 and 32×32. We also use the Discrete Cosine Transform (DCT) described in chapter 2 with only one RVM. To evaluate the performance of the network, we extracted two hundred and fifty sections of 128×128 from seismic time slices using the SEAM-II data set. Then, we randomly removed receivers at various percentages and reconstructed the signals using different configuration of parameters.

Figure 4.8 shows the mean Q against the computational time in seconds. By using 80% of the receivers, the accuracy is similar for all configurations of the RVM which use the Haar wavelet transform. This is due to the fact that there are enough training data. The reconstruction accuracy of the RVM with the DCT is significantly better. Smaller patches take less time to execute, along with architectures with fewer layers. By using 40%, there is a difference in accuracy between the first layer of the Haar wavelets and the rest. This is due to the fact that there are regions that are uncovered. The RVM with the DCT still provides better accuracy. By using 20% of the receivers, the RVM with one layer is again the worst as opposed to the RVM with DCT on 32×32 performing much better albeit being the slowest. All experiments were performed as single-core jobs on machines with Intel(R) Xeon(R) CPU E5-2650 with 2.00GHz.

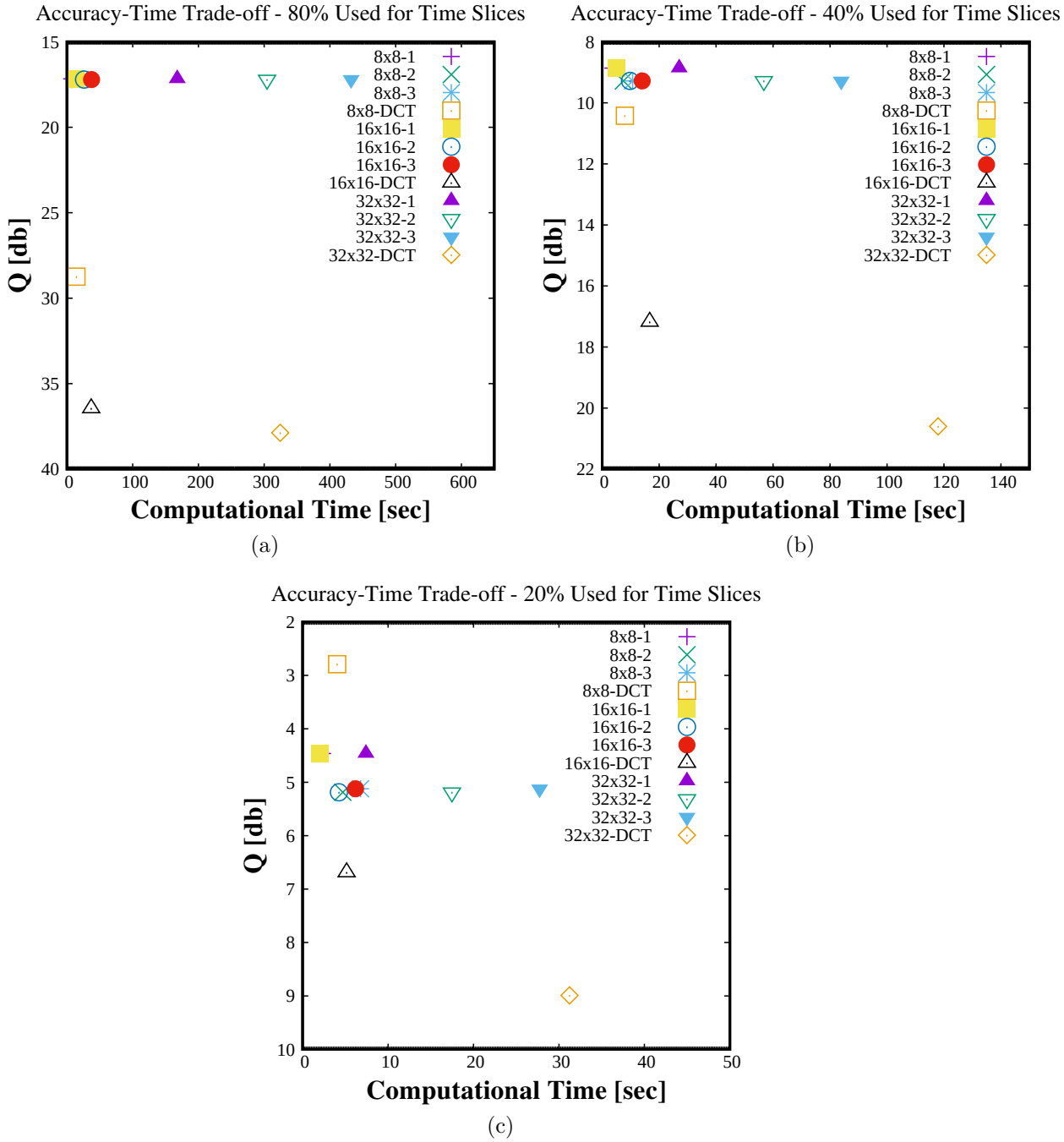


Fig. 4.8 Accuracy and computational time trade-off for various configurations of the network. Each configuration is labelled first with its patch size and then with the number of layers used. We show the trade-off when (a) 80%, (b) 40%, (c) 20% of receivers are used.

4.5 Reconstruction accuracy for time slices

We have seen that different configurations of the RVM can produce different results. For comparison purposes, we will use the best configuration with regards to the reconstruction accuracy, Q. This configuration is the RVM using the DCT bases and the larger the patch size the better. Since we will be operating in a 128×128 grid of receivers, the best possible patch size is 128×128 receivers. In all experiments, the Spectral Projected Gradient for L1 (SPGL1) package[1] with the Discrete Cosine Transform (DCT) is utilised. Similarly, for Projection Onto Convex Sets (POCS), the MATLAB code[2] was used and for the RVM the package from the author's website[3] was used.

Before performing the comparisons, we will also investigate various configurations for POCS and SPGL1. Different initialisation of parameters, patch sizes that they operate on, the choice of basis functions, stopping criteria for all algorithms, different thresholding operators for POCS to name a few, all can affect the results. In order to address this variability in full, experiments with all possible setups are necessary. However, in this thesis, we decided to explore two key elements, the patch size and the stopping criteria. We performed experiments in the time slice domain for POCS and SPGL1 to determine a suitable set of parameters. The choice of the dictionary for SPGL1 was fixed to the Discrete Cosine Transform (DCT). To ensure that the results are consistent over different instances of signals with different structures and variance, we have extracted two hundred and fifty sections of size 128×128 from time slices from the SEAM-II data set.

4.5.1 POCS configurations for time slices

We experimented with different number of iterations for the algorithm to terminate and the patch size was varied between the following: $\{8 \times 8, 16 \times 16, 32 \times 32, 64 \times 64, 128 \times 128\}$ and non-overlapping. A plot of mean Q against the measurements over all sections is given in Figure 4.9. It can be seen that the larger the patch size the better the reconstruction accuracy. The configuration with the best performance in our experiments operates on 128×128 patches, however the number of iterations is not obvious whether using 500 or 1000 iterations, the results are very similar with 1000 iterations being slightly better.

[1] van den Berg, E., and M. P. Friedlander, 2007 SPGL1: A solver for large-scale sparse reconstruction http://www.cs.ubc.ca/labs/scl/spgl1, accessed 4 May 2016.

[2] Projection Onto Convex Sets (POCS) software, http://www.freeusp.org/synthetics/POCS_example/, accessed 4 May 2016.

[3] Relevance Vector Machine (RVM) software, http://www.miketipping.com/downloads.htm, accessed 4 May 2016.

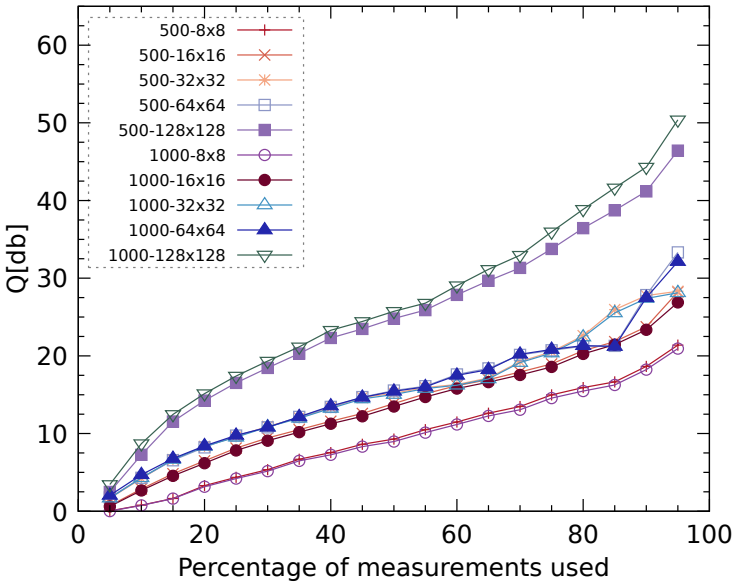


Fig. 4.9 Mean Q plot over two hundred and fifty sections of time slices of size 128×128 for different POCS configurations. We vary the number of iterations for termination (first number in the legend) and the patch size (second number in the legend).

Therefore, in the following comparisons, POCS with 500 iterations and a patch size of 128×128 will be used since the accuracy is similar to 1000 iterations but faster.

4.5.2 SPGL1 configurations for time slices

SPGL1 can be modified greatly with regards to its stopping criteria, and an exhaustive parameter search would be required. One stopping criterion checks the residual between the true available data and the reconstruction signal, another checks convergence of intermediate solvers, and another sets the maximum number of iterations. We experimented with the value of the residual and suggest using a difference which is much smaller than the l_2 norm of the available data, e.g. between $10^{-6}\|\mathbf{x}\|_2$ and $10^{-9}\|\mathbf{x}\|_2$. Figure 4.10 shows the mean Q with patch sizes from 8×8 to 128×128. The 128×128 patch size gives the best performance when less than 85% of the measurements are used, slightly better than 32×32 and much better than the rest. Larger patch sizes could perform better if the stopping criteria were tuned, i.e. by changing the number of iterations.

4.5.3 Comparisons against POCS and SPGL1

By using the insights from the experiments with different algorithmic configurations, we are now ready to compare the RVM against POCS and SPGL1. We ran experiments on

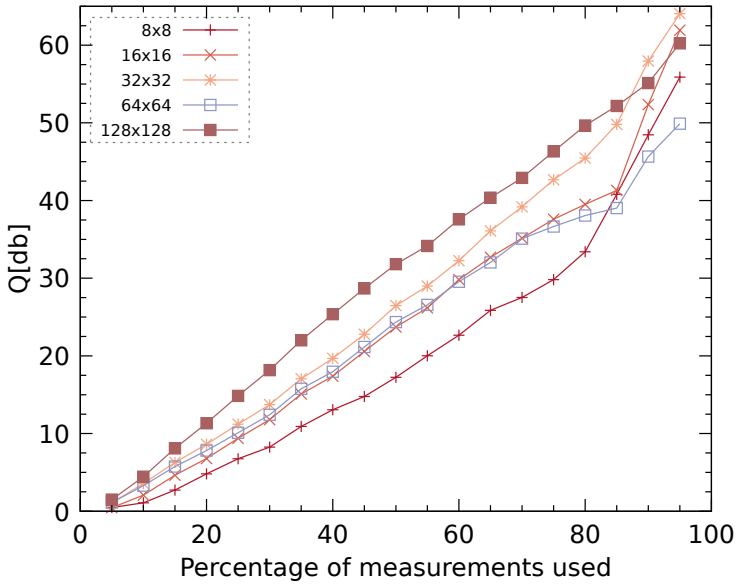


Fig. 4.10 Mean Q plot over two hundred and fifty sections of time slices of size 128×128 for different SPGL1 configurations. We vary the patch size (number in the legend).

a hundred and fifty sections of time slices extracted from the SEAM-II data set both close to the source and far from it (as illustrated in Figure 2.6), using 8×8 and 128×128 patch sizes. For the experiments on 8×8, we use overlapping patches (additional vertical and horizontal patches in the mid point of each patch) in order to avoid edge effects in reconstructions. This has the effect of slowing down the reconstruction but it will be useful when making comparisons in the next chapter (dictionaries of bases are learned only on 8×8 patches).

Two important criteria for each algorithm are: the reconstruction accuracy in Q and its computational time. We removed receivers randomly using different percentages and then used all algorithms to reconstruct the sections. The mean Q against the percentage of measurements is plotted in Figure 4.11. For POCS, SPGL1 and RVM on 8×8 patches, we used overlaps and ran experiments between 20% and 95% for every 5%. We can see that using 8×8 patches, the RVM with DCT bases gives the best performance in general. Then, the SPGL1 with DCT bases performs better using 40% of receivers and more compared to POCS and POCS performs better than SPGL1 with DCT when less than 40% of receivers are used.

The same behaviour is observed when using 128×128 patch size. For the RVM with 128×128 patch size, we ran experiments between 20% and 70% and every 10%. This is due to the fact that each run takes significant amount of computation as we will see later on. For POCS and SPGL1, experiments between 5% and 95% every 5% are undertaken.

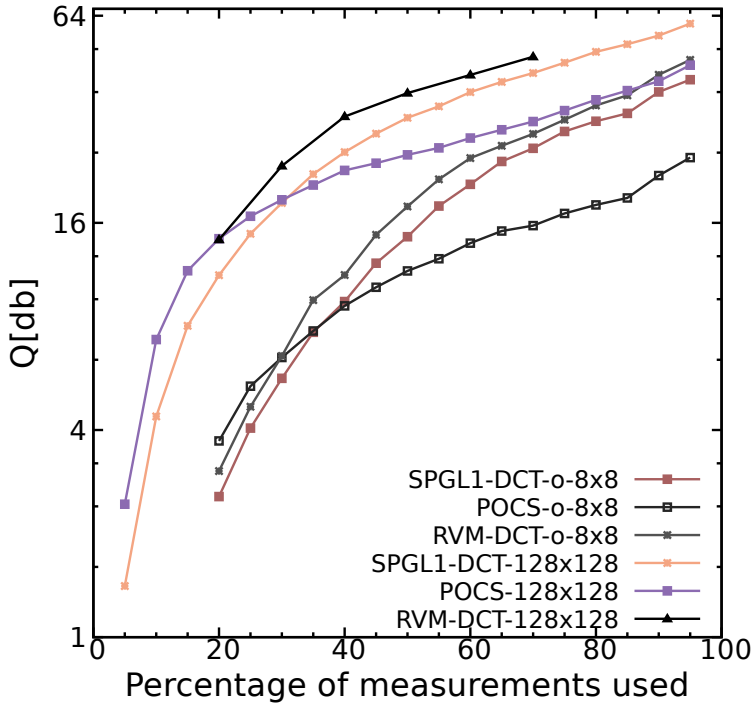
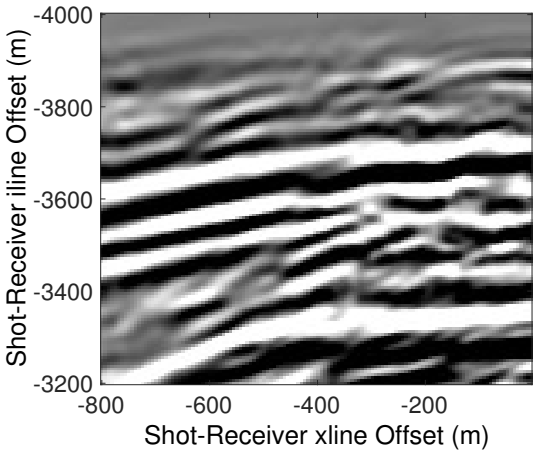


Fig. 4.11 Mean reconstruction accuracy, Q, against the percentage of measurements for one hundred and fifty seismic sections of time slices.

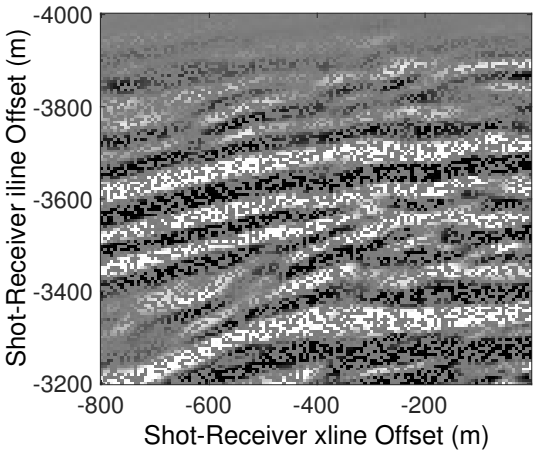
All algorithms perform better than their 8×8 configuration as expected. The RVM is still better than the rest, with SPGL1 performing better than POCS when using 35% of receivers and more. The overall trend is the same for 8×8 and 128×128 patches.

Reconstruction examples

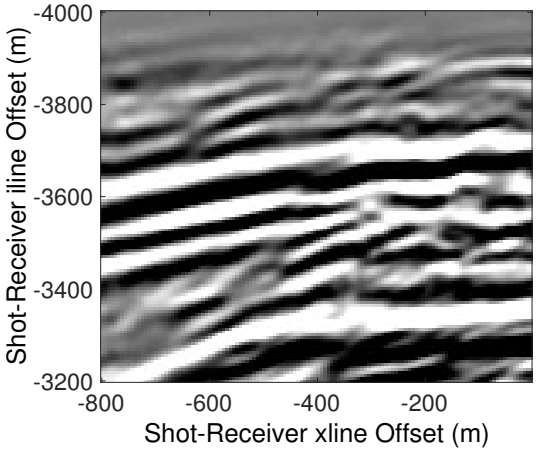
Two examples of reconstructions from sections of time slices are included in order to visualise the differences between algorithms. In this chapter, we will only illustrate examples with 128×128 patch sizes. In the next chapter, we will provide reconstructions using 8×8 patches as we will compare reconstructions with and without learned bases. Thus, for now, Figure 4.12 includes reconstructions using 128×128 patch size far from the source and Figure 4.13 includes reconstructions closer to the source. Figure 4.12(a) shows the original section and Figure 4.12(b) shows the same signal using only 50% of the receivers randomly. The reconstruction of the RVM using the DCT is in Figure 4.12(c), the SPGL1 with the same bases is in Figure 4.12(d) and the POCS in Figure 4.12(e). We can see that the reconstruction using the RVM is better compared to the rest of the algorithms. In addition, the reconstruction accuracy, Q, is higher for the RVM as opposed to the other algorithms.



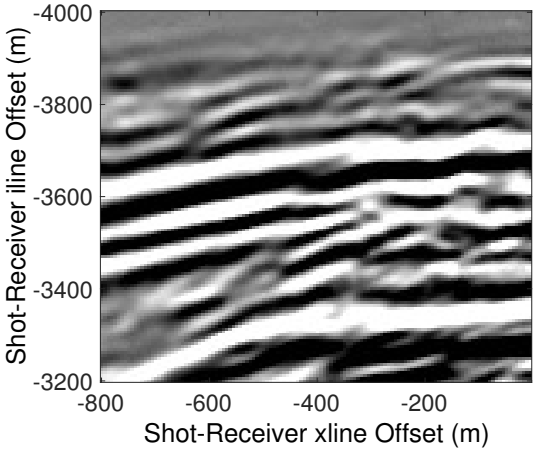
(a) Original



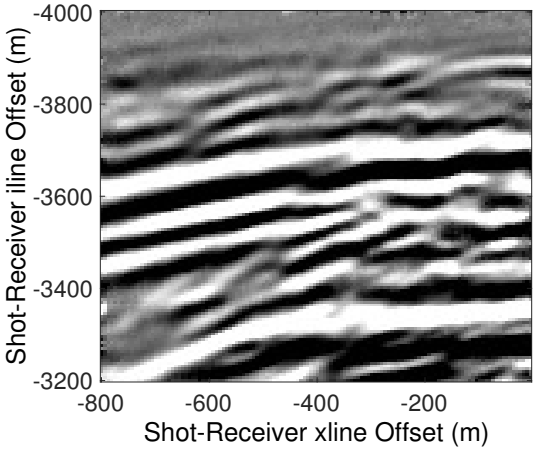
(b) 50% used



(c) RVM-DCT, 128×128, Q = **42.07** db

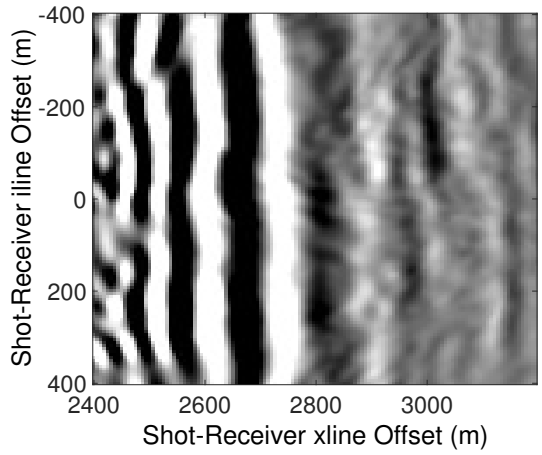


(d) SPGL1-DCT, 128×128, Q = 34.71 db

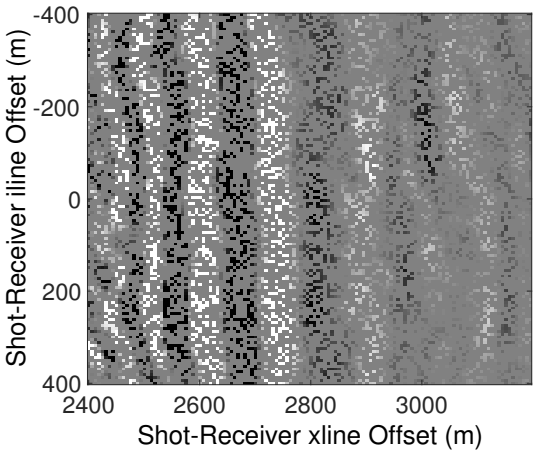


(e) POCS, 128×128, Q = 24.32 db

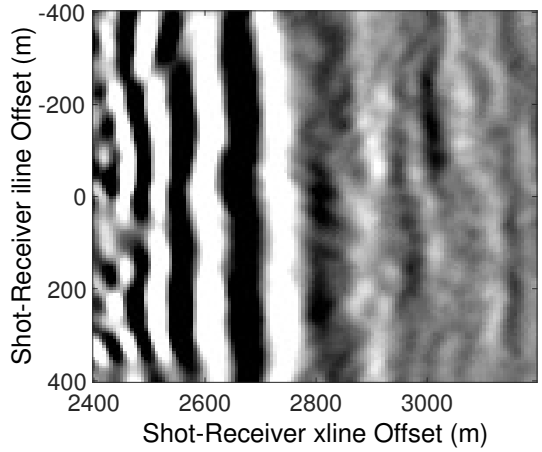
Fig. 4.12 A section from a time slice far from the source. Reconstructions with different algorithms are included using 50% of receivers.



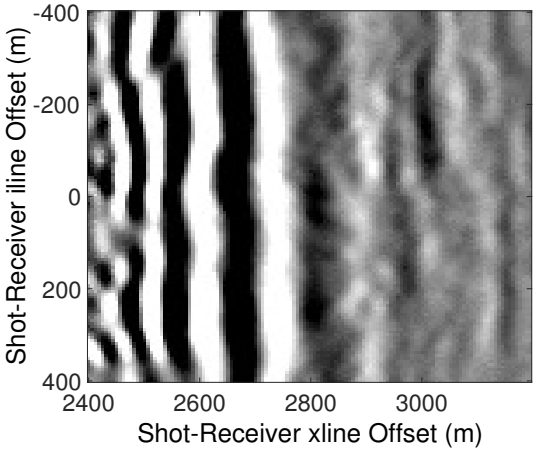
(a) Original



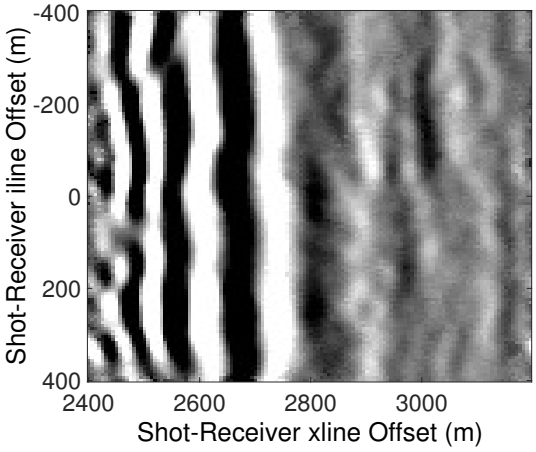
(b) 30% used



(c) RVM-DCT, 128×128, Q = **36.79** db

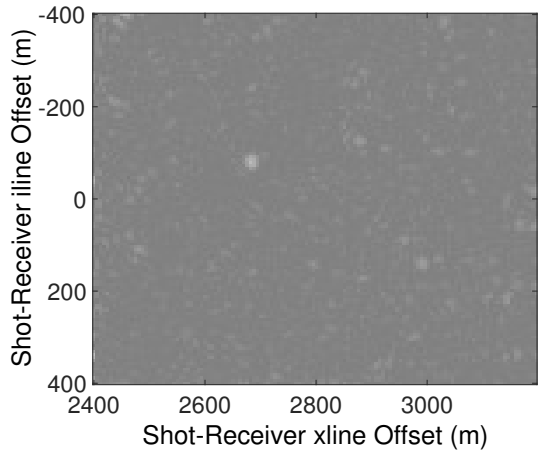


(d) SPGL1-DCT, 128×128, Q = 27.92 db

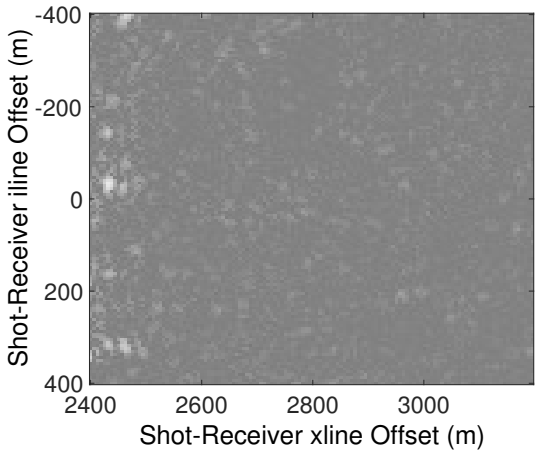


(e) POCS, 128×128, Q = 24.84 db

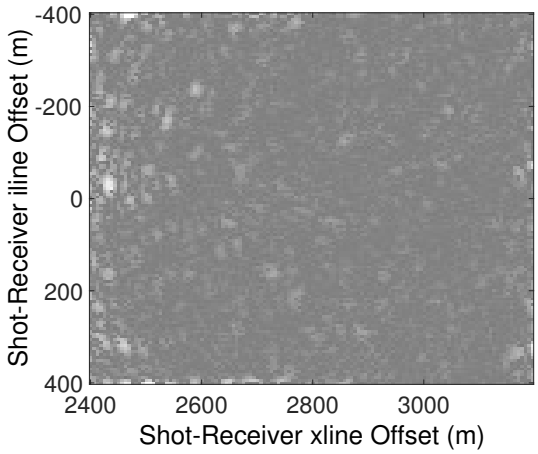
Fig. 4.13 A section from a time slice closer to the source. Reconstructions with different algorithms are included using 30% of receivers.



(a) RVM-DCT, 128 × 128, error map



(b) SPGL1-DCT, 128 × 128, error map



(c) POCS, 128 × 128, error map

Fig. 4.14 Reconstruction error maps for the reconstructions in Figure 4.13.

Figure 4.13(a) shows an original section from a time slice which includes receiver lines that pass close to the source. Figure 4.13(b) shows the same signal using only 30% of the receivers. Figure 4.13(c) shows the reconstruction of the RVM using DCT, Figure 4.13(d) shows the respective reconstruction of the SPGL1 with DCT and Figure 4.13(e) shows the reconstruction of POCS. We can see again that the RVM performs much better as opposed to the other algorithms. For these reconstructions, we have also included reconstruction error maps in Figure 4.14. From these, it can be seen that the error in the RVM in Figure 4.14(a) is small as opposed to the error maps of the SPGL1 in Figure 4.14(b) and POCS in Figure 4.14(c).

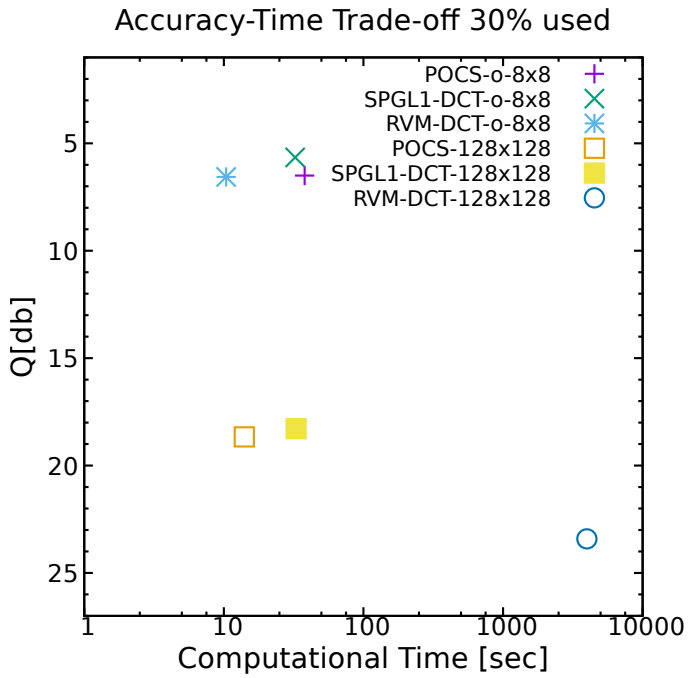


Fig. 4.15 Plot of mean reconstruction accuracy against mean computational time for one hundred and fifty sections of time slices using different algorithms with 30% of receivers.

Trade off plots of accuracy and time for all algorithms

Another important factor is the computational time that algorithms take for each reconstruction to complete. To test this, we recorded their computational time with all experiments again performed as single-core jobs on machines with Intel(R) Xeon(R) CPU E5-2650 with 2.00GHz. Figure 4.15 shows the mean Q against the mean time for one hundred and fifty sections of time slices using 30% of receivers. The RVM on 128×128 patch size obtains the best Q but also takes the longest time. POCS and SPGL1 on 128×128 obtain similar accuracy with SPGL1 being slower. On 8×8 overlapping patches, the behaviour in speed is opposite but all algorithms obtain similar Q. Figure 4.16 shows the same configurations but using 50% of receivers. The RVM on 128×128 patch size obtains the best accuracy but takes the longest time. SPGL1 is again slower than POCS but the difference in Q increases and is in favour of SPGL1. For 8×8 patches, there is a difference in Q. The RVM obtains better accuracy and is faster as opposed to the others. Finally, Figure 4.17 shows the same analysis using 70% of receivers. The RVM on 128×128 patches obtains the best accuracy and takes the longest time. The others have similar behaviour as before. The RVM on 8×8 patches obtains the highest reconstruction out of the 8×8 configurations.

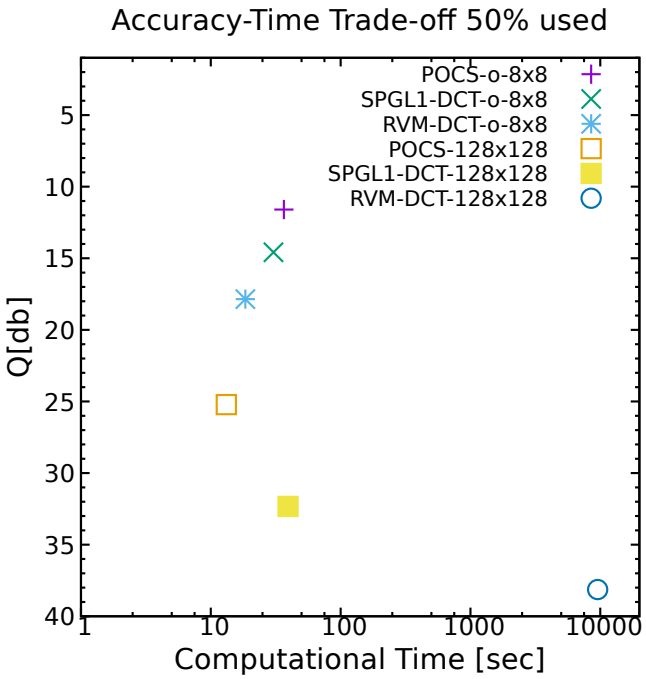


Fig. 4.16 Plot of mean reconstruction accuracy against mean computational time for one hundred and fifty sections of time slices using different algorithms with 50% of receivers.

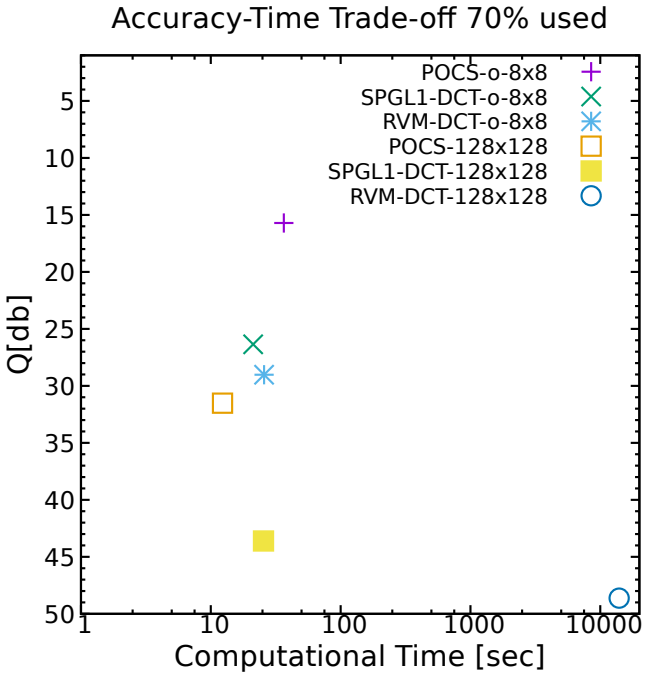
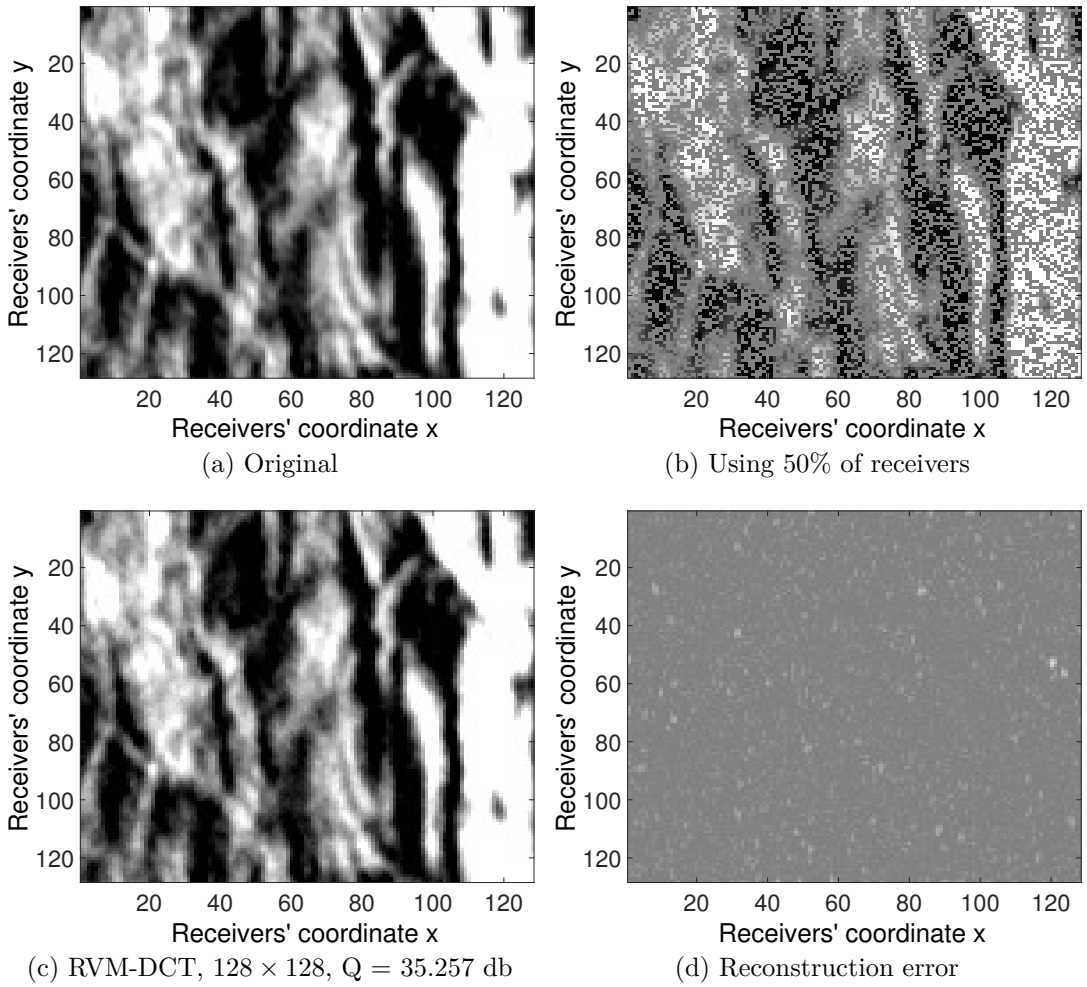


Fig. 4.17 Plot of mean reconstruction accuracy against mean computational time for one hundred and fifty sections of time slices using different algorithms with 70% of receivers.

4.6 Field data set experiment

We use the Parihaka data set (SEG, 2018b), a 3D seismic image provided by New Zealand Petroleum and Minerals (NZPM) mentioned in section 2.1 to test the RVM on field data. Figure 4.18(a) shows a section from a time slice of the data set. We removed 50% of the receivers randomly as shown in Figure 4.18(b) and reconstructed it using the RVM as seen in Figure 4.18(c). The reconstruction error can be seen in Figure 4.18(d) with small differences. This illustrates the reconstruction of field data using the RVM with fixed dictionary of bases, the Discrete Cosine Transform (DCT). Nevertheless, fixing the dictionary is limiting and we thus propose another algorithm in the next chapter that is able to learn the dictionary from available data.



(a) Original

(b) Using 50% of receivers

(c) RVM-DCT, 128×128, Q = 35.257 db

(d) Reconstruction error

Fig. 4.18 A section from a time slice of the Parihaka field data set. (a) shows the original, (b) shows 50% of receivers, (c) shows the reconstruction and the error in (d).

Chapter 5

Beta Process Factor Analysis for Seismic Compressive Sensing

Most of the methods in the seismic Compressive Sensing (CS) literature use predefined dictionaries of basis functions with a fixed size which is very limiting. There are many possibilities for the dictionaries to be used (refer to section 2.5 for seismic specific bases) with potentially different reconstruction accuracy. Experimenting with all the choices of basis functions is not practical. On the contrary, learning a dictionary of bases from the available data is advantageous since it can adapt to the available data. In this chapter, we apply Beta Process Factor Analysis (BPFA)(Paisley and Carin, 2009; Zhou et al., 2012) to seismic signals and learn dictionaries of bases for the purpose of CS and denoising. We start the discussion with a description of the BPFA model. Then, we use BPFA to learn sparse representations of seismic signals in the time slice domain. Various experiments and comparisons against POCS, SPGL1 and RVM are undertaken. We propose two hybrid algorithms that use the BPFA bases to obtain higher reconstruction accuracy and faster computational time. We also provide a Gibbs analysis and utilise its insights to speed up the BPFA inference further. Other tests with missing blocks and artificial rivers are undertaken along with examples of 3D reconstruction and denoising.

5.1 The BPFA model

BPFA is a hierarchical Bayesian model that constitutes a finite approximation to the Indian Buffet Process (IBP) (Griffiths and Ghahramani, 2011). It is a truncated beta-Bernoulli process with a fixed, large number of features, L, and shrinks if there is redundancy. To introduce this model, we assume that a data matrix $\mathbf{X}$ is generated by

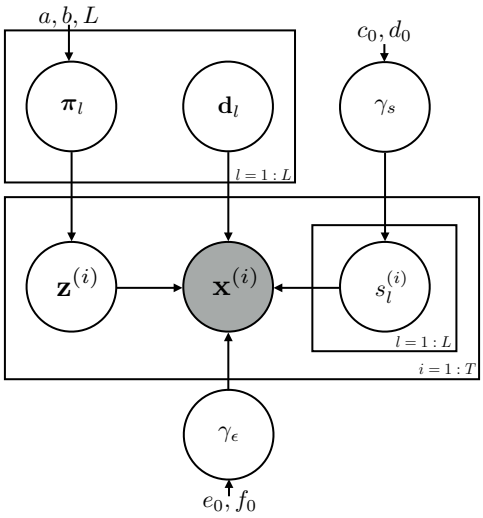


Fig. 5.1 Graphical model of the Beta Process Factor Analysis. The circles represent the random variables of the model that are described by probability distributions. The others are the parameters which govern each probability distribution.

an underlying process with its columns $\mathbf{x}^{(i)} \in \mathbb{R}^K$, $i = 1, ..., T$ generated by the graphical model in Figure 5.1. Each $\mathbf{x}^{(i)}$ can be considered a patch from a section of a time slice and the collection of patches could be extracted from one section. This means that $M = K * T$ is the number of the original receivers and from the definition of equation 3.30, as a reminder N is the size of only the available receivers with $N \ll M$. We want to model each patch at its original domain size in order to learn bases of that dimension and use a mask to indicate missing receivers (see equation 5.10).

The *likelihood function* for the BPFA model is given by

$$\mathbf{x}^{(i)} = \mathbf{D}\mathbf{w}^{(i)} + \boldsymbol{\epsilon}^{(i)}, \tag{5.1}$$

where $\mathbf{D}$ denotes the dictionary of bases to be learned, defined as $\mathbf{D} \in \mathbb{R}^{K \times L}$. This illustrates that the input space is different from the previous chapter since we split the sections in smaller patches. $\mathbf{w}^{(i)} \in \mathbb{R}^L$ are the coefficients of the model and $\boldsymbol{\epsilon}^{(i)} \in \mathbb{R}^K$ is assumed Gaussian noise. The likelihood function is thus a normal distribution. In the Bayesian framework, *prior distributions* are defined for each model variable. We start with the columns $\{\mathbf{d}_l\}_{l=1}^L$ of $\mathbf{D}$ which are modelled by

$$p(\mathbf{d}_l) = \mathcal{N}(\mathbf{d}_l; \mathbf{0}, K^{-1}\mathbf{I}_K) \tag{5.2}$$

where $\mathbf{I}_K \in \mathbb{R}^{K\times K}$ is the identity matrix. Then, we explicitly separate the value of a coefficient in $\mathbf{w}^{(i)}$ from the fact whether it is non-zero or zero. This means that if the coefficient is non-zero, the corresponding basis is used when generating $\mathbf{x}^{(i)}$. In particular, we introduce latent variables $\mathbf{z}^{(i)}$ and $\mathbf{s}^{(i)}$ such that

$$\mathbf{w}^{(i)} = \mathbf{z}^{(i)} \odot \mathbf{s}^{(i)}, \tag{5.3}$$

where $\odot$ represents the elementwise vector product, $\mathbf{z}^{(i)} \in \{0,1\}^L$ signifies whether a basis is used and $\mathbf{s}^{(i)} \in \mathbb{R}^L$ are the values of the coefficients. The prior distribution for $\mathbf{z}^{(i)}$ is given by Bernoulli distributions,

$$p(\mathbf{z}^{(i)}) = \prod_{l=1}^{L} \text{Bernoulli}(z_l^{(i)}; \pi_l), \tag{5.4}$$

where π_l is the probability that the l-th basis is used when $\mathbf{x}^{(i)}$ is generated. This means that the l-th component of $\mathbf{z}^{(i)}$ is generated by a Bernoulli distribution with probability π_l. The probabilities $\boldsymbol{\pi} = [\pi_1, ..., \pi_L]^T$ themselves are a priori distributed by a hyper-prior defined by,

$$p(\boldsymbol{\pi}) = \prod_{l=1}^{L} \text{Beta}(\pi_l; a/L, b(L-1)/L), \tag{5.5}$$

where a, b are parameters characterising the distribution.

The latent variable, $\mathbf{s}^{(i)}$, models the value of the coefficients and is assumed to be generated by,

$$p(\mathbf{s}^{(i)}) = \mathcal{N}(\mathbf{s}^{(i)}; \mathbf{0}, \gamma_s^{-1}\mathbf{I}_L) \tag{5.6}$$

where $\mathbf{I}_L$ is the $L \times L$ identity matrix and γ_s is modelled in turn by a hyper prior,

$$p(\gamma_s) = \text{Gamma}(\gamma_s; c_0, d_0). \tag{5.7}$$

Finally, the noise, $\boldsymbol{\epsilon}^{(i)}$ is modelled by,

$$p(\boldsymbol{\epsilon}^{(i)}) = \mathcal{N}(\boldsymbol{\epsilon}^{(i)}; \mathbf{0}, \gamma_\epsilon^{-1}\mathbf{I}_K), \tag{5.8}$$

where $\mathbf{I}_K$ is the $K \times K$ identity matrix and γ_ϵ is modelled by,

$$p(\gamma_\epsilon) = \text{Gamma}(\gamma_\epsilon; e_0, f_0). \tag{5.9}$$

All these probability distributions and their parameters are summarised in Figure 5.1 and we will discuss their settings later.

In the case of training the BPFA model with missing data, a sampling matrix (mask) is required. We will denote this as $\mathbf{\Delta}^{(i)} \in \{0,1\}^{m \times K}$ constructed by removing rows from the identity matrix of the corresponding missing locations in $\mathbf{x}^{(i)}$ and m being the number of available receivers in a patch. We will denote $\mathbf{y}^{(i)} \in \mathbb{R}^{\|\mathbf{\Delta}^{(i)}\|_0}$ and given by

$$\mathbf{y}^{(i)} = \mathbf{\Delta}^{(i)}\mathbf{x}^{(i)} = \mathbf{\Delta}^{(i)}\mathbf{D}(\mathbf{s}^{(i)} \odot \mathbf{z}^{(i)}) + \mathbf{\Delta}^{(i)}\boldsymbol{\epsilon}^{(i)}. \tag{5.10}$$

Note that in practice, we do not collapse the signal as in the RVM and SPGL1 but rather insert zeros at the missing locations (similar to POCS). This is achieved by using each $\mathbf{\Delta}^{(i)}$ to project the data onto the original domain, acting as operators to indicate the location of the available data. Using the likelihood and prior distributions, we obtain a joint distribution for the model. We define $\mathbf{Y} = [\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, ..., \mathbf{y}^{(T)}]$, $\mathbf{\Delta} = [\mathbf{\Delta}^{(1)}, \mathbf{\Delta}^{(2)}, ..., \mathbf{\Delta}^{(T)}]$, $\mathbf{Z} = [\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, ..., \mathbf{z}^{(T)}]$ and $\mathbf{S} = [\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, ..., \mathbf{s}^{(T)}]$. Thus, the joint distribution is given by

$$\begin{aligned} P(\mathbf{D}, \mathbf{Z}, \mathbf{S}, \boldsymbol{\pi}, \gamma_\epsilon, \gamma_s, \mathbf{Y}, \mathbf{\Delta}) = &\prod_{i=1}^{T} \mathcal{N}(\mathbf{y}^{(i)}; \mathbf{\Delta}^{(i)}\mathbf{D}(\mathbf{s}^{(i)} \odot \mathbf{z}^{(i)}), \gamma_\epsilon^{-1}\mathbf{I}_{\|\mathbf{\Delta}^{(i)}\|_0}) \mathcal{N}(\mathbf{s}^{(i)}; \mathbf{0}, \gamma_s^{-1}\mathbf{I}_L) \\ &\prod_{l=1}^{L} \mathcal{N}(\mathbf{d}_l; \mathbf{0}, K^{-1}\mathbf{I}_K) \text{Beta}(\pi_l; \frac{a}{L}, \frac{b(L-1)}{L}) \\ &\prod_{i=1}^{T}\prod_{l=1}^{L} \text{Bernoulli}(z_l^{(i)}; \pi_l) \text{Gamma}(\gamma_s; c_0, d_0) \text{Gamma}(\gamma_\epsilon; e_0, f_0). \end{aligned} \tag{5.11}$$

Summary of BPFA inference

We have discussed all the modelling assumptions incorporated via the likelihood function and prior distributions. To obtain the conditional posterior distributions for the model variables, we write the expressions for the likelihood and the prior distribution of each variable. Multiplying out and simplifying expressions, we obtain closed-form distribution due to conjugacy. For their detailed derivations, refer to the Appendix A of Dang (2016). We provide the conditional posterior distributions for each model variable (Zhou et al., 2012) that is used by Gibbs sampling to obtain the variables of interest.

Gibbs sampling update equation for bases dictionary

The prior distribution and the likelihood function are both normal distributions,

$$p(\mathbf{d}_l | \mathbf{D}_{-l}, \mathbf{Z}, \mathbf{S}, \boldsymbol{\pi}, \gamma_\epsilon, \gamma_s, \mathbf{Y}, \mathbf{\Delta}) \propto \underbrace{\prod_{i=1}^{T} \mathcal{N}(\mathbf{y}^{(i)}; \mathbf{\Delta}^{(i)}\mathbf{D}(\mathbf{s}^{(i)} \odot \mathbf{z}^{(i)}), \gamma_\epsilon^{-1}\mathbf{I}_{\|\mathbf{\Delta}^{(i)}\|_0})}_{\text{Likelihood of data}} \overbrace{\mathcal{N}(\mathbf{d}_l; \mathbf{0}, K^{-1}\mathbf{I}_K)}^{\text{Prior of basis}}$$

and thus, we expect the posterior distribution of $\mathbf{d}_l$ to be a normal distribution as well, due to conjugacy. This is given by

$$p(\mathbf{d}_l|\mathbf{D}_{-l}, \mathbf{Z}, \mathbf{S}, \boldsymbol{\pi}, \gamma_\epsilon, \gamma_s, \mathbf{Y}, \boldsymbol{\Delta}) \propto \mathcal{N}(\boldsymbol{\mu}_{\mathbf{d}_l}, \boldsymbol{\Sigma}_{\mathbf{d}_l}), \tag{5.12}$$

where,

$$\boldsymbol{\Sigma}_{\mathbf{d}_l} = \left(K\mathbf{I}_K + \gamma_\epsilon \sum_{i=1}^{T} (w_l^{(i)})^2 (\boldsymbol{\Delta}^{(i)})^T \boldsymbol{\Delta}^{(i)} \right)^{-1} \tag{5.13}$$

and

$$\boldsymbol{\mu}_{\mathbf{d}_l} = \gamma_\epsilon \boldsymbol{\Sigma}_{\mathbf{d}_l} \sum_{i=1}^{T} w_l^{(i)} \tilde{\mathbf{x}}_{-l}^{(i)} \tag{5.14}$$

with

$$\tilde{\mathbf{x}}_{-l}^{(i)} = (\boldsymbol{\Delta}^{(i)})^T \mathbf{y}^{(i)} - (\boldsymbol{\Delta}^{(i)})^T \boldsymbol{\Delta}^{(i)} \sum_{\substack{j=1 \\ j \neq l}}^{L} w_j^{(i)} \mathbf{d}_j. \tag{5.15}$$

$\tilde{\mathbf{x}}_{-l}^{(i)}$ will be used throughout the equations and is the error the current model makes explaining $\mathbf{x}^{(i)}$ when the basis $\mathbf{d}_l$ is not used. The mean, $\boldsymbol{\mu}_{\mathbf{d}_l}$, can be seen to be the weighted sum of the errors skewed by $\gamma_\epsilon \boldsymbol{\Sigma}_{\mathbf{d}_l}$.

Gibbs sampling update equation for binary indicators

The posterior distribution of $z_l^{(i)}$ is the product of the likelihood function and the prior distribution on the binary indicators. This is given by,

$$p(z_l^{(i)}|\mathbf{D}, \mathbf{Z}_{-li}, \mathbf{S}, \boldsymbol{\pi}, \gamma_\epsilon, \mathbf{Y}, \boldsymbol{\Delta}) \propto \underbrace{\mathcal{N}(\mathbf{y}^{(i)}; \boldsymbol{\Delta}^{(i)} \mathbf{D}(\mathbf{s}^{(i)} \odot \mathbf{z}^{(i)}), \gamma_\epsilon^{-1} \mathbf{I}_{\|\boldsymbol{\Delta}^{(i)}\|_0})}_{\text{Likelihood of data}} \overbrace{\text{Bernoulli}(z_l^{(i)}; \pi_l)}^{\text{Prior of indicators}}. \tag{5.16}$$

The posterior distribution is given by,

$$p(z_l^{(i)}|\mathbf{D}, \mathbf{Z}_{-li}, \mathbf{S}, \boldsymbol{\pi}, \gamma_\epsilon, \mathbf{Y}, \boldsymbol{\Delta}) \propto \text{Bernoulli}\left(\frac{\text{Prob\_on}}{\text{Prob\_off} + \text{Prob\_on}} \right), \tag{5.17}$$

where the posterior probability that $z_l^{(i)} = 1$ is proportional to,

$$\text{Prob\_on} = \pi_l \exp\left\{ -\frac{\gamma_\epsilon}{2} \left((s_l^{(i)})^2 \mathbf{d}_l^T (\boldsymbol{\Delta}^{(i)})^T \boldsymbol{\Delta}^{(i)} \mathbf{d}_l - 2 s_l^{(i)} \mathbf{d}_l^T \tilde{\mathbf{x}}_{-l}^{(i)} \right) \right\}. \tag{5.18}$$

In the same manner, using the probability of $1-\pi_l$ of $z_l^{(i)}=0$ happening, the posterior probability of $z_l^{(i)}=0$ is given by

$$\text{Prob\_off} = 1-\pi_l. \tag{5.19}$$

Note that in equation 5.18, $\mathbf{d}_l^T\tilde{\mathbf{x}}_{-l}^{(i)}$ gives the alignment of basis $\mathbf{d}_l$ with the error when not using $\mathbf{d}_l$. If it is close to zero (orthogonal), the probability that the basis is used is reduced.

Gibbs sampling update equation for the coefficients' value

The posterior distribution of $s_l^{(i)}$ is the product of two normal distributions and from conjugacy we expect this to be a normal distribution as well,

$$p(s_l^{(i)}|\mathbf{D},\mathbf{S}_{-li},\mathbf{Z},\boldsymbol{\pi},\gamma_\epsilon,\gamma_s,\mathbf{Y},\boldsymbol{\Delta}) \propto \underbrace{\mathcal{N}(\mathbf{y}^{(i)};\boldsymbol{\Delta}^{(i)}\mathbf{D}(\mathbf{s}^{(i)}\odot\mathbf{z}^{(i)}),\gamma_\epsilon^{-1}\mathbf{I}_{\|\boldsymbol{\Delta}^{(i)}\|_0})}_{\text{Likelihood of data}} \overbrace{\mathcal{N}(\mathbf{s}^{(i)};0,\gamma_s^{-1}\mathbf{I}_L)}^{\text{Prior on coefficients' value}}$$

This is given by

$$p(s_l^{(i)}|\mathbf{D},\mathbf{S}_{-li},\mathbf{Z},\boldsymbol{\pi},\gamma_\epsilon,\gamma_s,\mathbf{Y},\boldsymbol{\Delta}) \propto \mathcal{N}(\mu_{s_l^{(i)}},\Sigma_{s_l^{(i)}}) \tag{5.20}$$

where

$$\Sigma_{s_l^{(i)}} = \begin{cases} \left(\gamma_s+\gamma_\epsilon\mathbf{d}_l^T(\boldsymbol{\Delta}^{(i)})^T\boldsymbol{\Delta}^{(i)}\mathbf{d}_l\right)^{-1} & \text{if } z_l^{(i)}=1 \\ \gamma_s^{-1} & \text{if } z_l^{(i)}=0 \end{cases} \tag{5.21}$$

and

$$\mu_{s_l^{(i)}} = \begin{cases} \gamma_\epsilon\Sigma_{s_l^{(i)}}\mathbf{d}_l^T\tilde{\mathbf{x}}_{-l}^{(i)} & \text{if } z_l^{(i)}=1 \\ 0 & \text{if } z_l^{(i)}=0 \end{cases} \tag{5.22}$$

Gibbs sampling update equation for the probabilities of sparsity

The posterior distribution of π_l is the product of a Beta and Bernoulli distributions and from conjugacy we expect this to be a Beta distribution as well,

$$p(\pi_l|\mathbf{D},\mathbf{Z},\mathbf{S},\boldsymbol{\pi}_{-l},\gamma_\epsilon,\gamma_s,\mathbf{Y},\boldsymbol{\Delta}) \propto \underbrace{\prod_{i=1}^{T}\text{Bernoulli}(z_l^{(i)};\pi_l)}_{\text{Likelihood of binary indicators}} \overbrace{\text{Beta}\left(\pi_l;\frac{a}{L},\frac{b(L-1)}{L}\right)}^{\text{Prior on sparsity}} \tag{5.23}$$

This is given by,

$$p(\pi_l|\mathbf{D},\mathbf{Z},\mathbf{S},\boldsymbol{\pi}_{-\pi_l},\gamma_\epsilon,\gamma_s,\mathbf{Y},\boldsymbol{\Delta}) \propto \text{Beta}\left(\frac{a}{L}+\sum_{i=1}^{T} z_l^{(i)}, \frac{b(L-1)}{L}+T-\sum_{i=1}^{T} z_l^{(i)}\right). \quad (5.24)$$

Gibbs sampling update equation for precision of coefficients' value

The posterior distribution of γ_s is the product of a normal distribution and a Gamma distribution which models the inverse of the variance (precision) of the normal. Thus, we expect a Gamma distribution as the posterior due to conjugacy,

$$p(\gamma_s|\mathbf{D},\mathbf{Z},\mathbf{S},\boldsymbol{\pi},\gamma_\epsilon,\mathbf{Y},\boldsymbol{\Delta}) \propto \underbrace{\prod_{i=1}^{T}\mathcal{N}(\mathbf{s}^{(i)};0,\gamma_s^{-1}\mathbf{I}_L)}_{\text{Likelihood of coefficients}} \overbrace{\text{Gamma}(\gamma_s;c_0,d_0)}^{\text{Prior for precision}} \quad (5.25)$$

This is given by,

$$p(\gamma_s|\mathbf{D},\mathbf{Z},\mathbf{S},\boldsymbol{\pi},\gamma_\epsilon,\mathbf{Y},\boldsymbol{\Delta}) \propto \text{Gamma}\left(c_0+\frac{TL}{2}, d_0+\frac{1}{2}\sum_{i=1}^{T}(\mathbf{s}^{(i)})^T\mathbf{s}^{(i)}\right) \quad (5.26)$$

Gibbs sampling update equation for precision of noise

The posterior distribution of γ_ϵ is the product of a normal distribution and a Gamma distribution which models the inverse of the variance (precision) of the normal. Thus, we expect a Gamma distribution as the posterior due to conjugacy as well given by,

$$p(\gamma_\epsilon|\mathbf{D},\mathbf{Z},\mathbf{S},\boldsymbol{\pi},\gamma_s,\mathbf{Y},\boldsymbol{\Delta}) \propto \underbrace{\prod_{i=1}^{T}\mathcal{N}(\mathbf{y}^{(i)};\boldsymbol{\Delta}^{(i)}\mathbf{D}\mathbf{w}^{(i)},\gamma_\epsilon^{-1}\mathbf{I}_{\|\boldsymbol{\Delta}^{(i)}\|_0})}_{\text{Likelihood of data}} \overbrace{\text{Gamma}(\gamma_\epsilon;e_0,f_0)}^{\text{Prior for precision}} \quad (5.27)$$

This is given by,

$$p(\gamma_\epsilon|\mathbf{D},\mathbf{Z},\mathbf{S},\boldsymbol{\pi},\gamma_s,\mathbf{Y},\boldsymbol{\Delta}) \propto \text{Gamma}\left(e_0+\frac{1}{2}\sum_{i=1}^{T}\|\boldsymbol{\Delta}^{(i)}\|_0, f_0+\frac{1}{2}\sum_{i=1}^{T}\|\mathbf{y}^{(i)}-\boldsymbol{\Delta}^{(i)}\mathbf{D}\mathbf{w}^{(i)})\|_2^2\right). \quad (5.28)$$

BPFA Algorithm

Using these posterior distributions, it is possible to create an algorithm for inference. BPFA estimates the model's variables using these expressions depending on other variables that are considered fixed for a given iteration. This is called a *Gibbs iteration* and the entire procedure of obtaining an estimation for one variable (using the posterior conditional distribution) given the others is *Gibbs sampling* which was described in subsection 3.2.2.

The summary of the BPFA inference is given in Algorithm 4 with discussions on initialisation given in sections 5.3 and 5.4. We dropped the dependence on $\mathbf{Y}$ and $\mathbf{\Delta}$ since these are the same for every iteration and for every distribution.

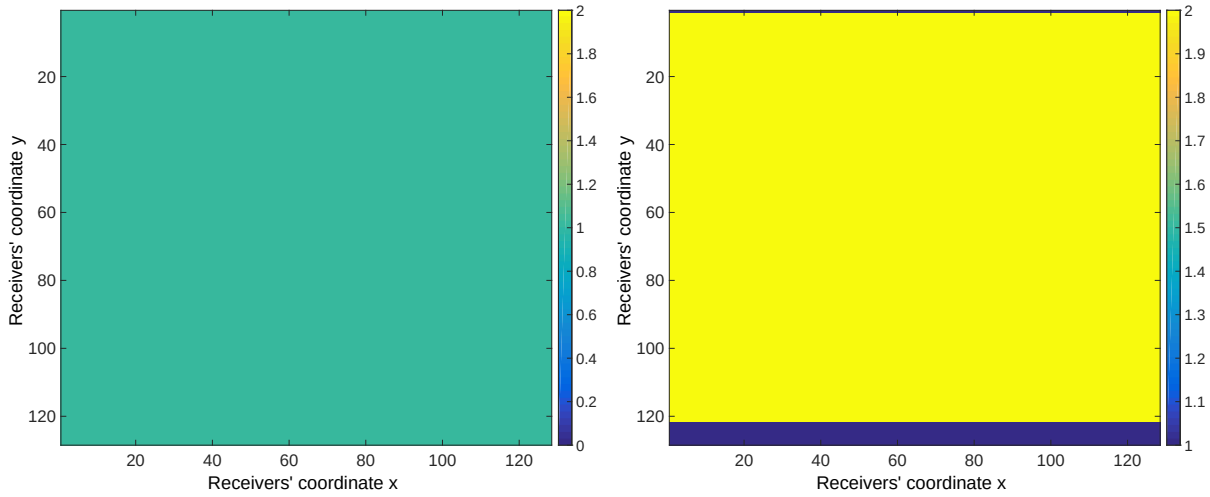
There is an aspect of the algorithm that we have not mentioned yet, namely the patch processing of the BPFA. As it can be seen in Algorithm 4, there are two big loops for R= 0 :Rounds and it= 0 :Iterations_R. These control which patches are to be processed at a particular instance and we will describe the patch processing next.

Algorithm 4 Beta Process Factor Analysis (BPFA)

Initialisation: $L, \mathbf{Z}^{(0,\,0)}, \mathbf{D}^{(0,\,0)}, \mathbf{S}^{(0,\,0)}, \boldsymbol{\pi}^{(0,\,0)}, \gamma_s^{(0,\,0)}, \gamma_\epsilon^{(0,\,0)}$
for `R=0:Rounds` **do**
  for `it=0:Iterations_R` **do**
    for `l=1:L` **do**
      $\mathbf{d}_l^{(\mathrm{R+1,\ it+1})} \sim p(\mathbf{d}_l|\mathbf{D}_{-l}^{(\mathrm{R,\ it})}, \mathbf{Z}^{(\mathrm{R,\ it})}, \mathbf{S}^{(\mathrm{R,\ it})}, \boldsymbol{\pi}^{(\mathrm{R,\ it})}, \gamma_\epsilon^{(\mathrm{R,\ it})}, \gamma_s^{(\mathrm{R,\ it})})$
    end for
    for `l=1:L` **do**
      for `i=1:T` **do**
        $\{z_l^{(i)}\}^{(\mathrm{R+1,\ it+1})} \sim p(z_l^{(i)}|\mathbf{D}^{(\mathrm{R+1,\ it+1})}, \mathbf{Z}_{-li}^{(\mathrm{R,\ it})}, \mathbf{S}^{(\mathrm{R,\ it})}, \boldsymbol{\pi}^{(\mathrm{R,\ it})}, \gamma_\epsilon^{(\mathrm{R,\ it})})$
      end for
    end for
    for `l=1:L` **do**
      for `i=1:T` **do**
    $\{s_l^{(i)}\}^{(\mathrm{R+1,\ it+1})} \sim p(s_l^{(i)}|\mathbf{D}^{(\mathrm{R+1,\ it+1})}, \mathbf{S}_{-li}^{(\mathrm{R,\ it})}, \mathbf{Z}^{(\mathrm{R+1,\ it+1})}, \boldsymbol{\pi}^{(\mathrm{R,\ it})}, \gamma_\epsilon^{(\mathrm{R,\ it})}, \gamma_s^{(\mathrm{R,\ it})})$
      end for
    end for
    for `l=1:L` **do**
    $\pi_l^{(\mathrm{R+1,\ it+1})} \sim p(\pi_l|\mathbf{D}^{(\mathrm{R+1,\ it+1})}, \mathbf{Z}^{(\mathrm{R+1,\ it+1})}, \mathbf{S}^{(\mathrm{R+1,\ it+1})}, \boldsymbol{\pi}_{-\pi_l}^{(\mathrm{R,\ it})}, \gamma_\epsilon^{(\mathrm{R,\ it})}, \gamma_s^{(\mathrm{R,\ it})})$
    end for
  $\gamma_s^{(\mathrm{R+1,\ it+1})} \sim p(\gamma_s|\mathbf{D}^{(\mathrm{R+1,\ it+1})}, \mathbf{Z}^{(\mathrm{R+1,\ it+1})}, \mathbf{S}^{(\mathrm{R+1,\ it+1})}, \boldsymbol{\pi}^{(\mathrm{R+1,\ it+1})}, \gamma_\epsilon^{(\mathrm{R,\ it})})$
  $\gamma_\epsilon^{(\mathrm{R+1,\ it+1})} \sim p(\gamma_\epsilon|\mathbf{D}^{(\mathrm{R+1,\ it+1})}, \mathbf{Z}^{(\mathrm{R+1,\ it+1})}, \mathbf{S}^{(\mathrm{R+1,\ it+1})}, \boldsymbol{\pi}^{(\mathrm{R+1,\ it+1})}, \gamma_s^{(\mathrm{R+1,\ it+1})})$
  end for
end for
return $\mathbf{D}^{(\mathrm{Rounds,\ Iterations\_R})}, \mathbf{Z}^{(\mathrm{Rounds,\ Iterations\_R})}, \mathbf{S}^{(\mathrm{Rounds,\ Iterations\_R})}, \gamma_\epsilon^{(\mathrm{Rounds,\ Iterations\_R})}$
end

5.2 Patch processing for BPFA

In order to obtain training data, we split a signal into smaller overlapping patches and use them in a sequential manner. In all subsequent BPFA experiments, we chose a patch size



(a) First round. (b) Second round.

Fig. 5.2 Heat map (a) of number of times each receiver is used in the section during the first Gibbs round. (b) shows the respective heat map for the second Gibbs round where some receivers are included in two patches and some in only 1.

of 8×8 in sections of time slices of 128×128. At the beginning, the algorithm extracts all 256 patches (there are $\frac{128}{8} = 16$ along the horizontal axis and $\frac{128}{8} = 16$ patches along the vertical axis). This is essentially what an algorithm would extract without overlaps. We call this the first round (i.e. first extraction of patches) and each round could have many Gibbs iterations. Then, in the second round, the algorithm shifts the starting point of extraction by one receiver down. This results in the extraction of 240 patches (there are again 16 along the horizontal axis and $16 - 1 = 15$ patches along the vertical axis since the last one is not a complete 8×8 patch). This procedure continues for all 64 receiver locations in a given patch, resulting in 64 such sets (or rounds) of patches where each set contains hundreds of patches.

For the first Gibbs round of the algorithm, the inference starts using the first set of patches. Then, in the second round the second set is extracted and then both first and second sets are used for estimation. Figure 5.2(a) shows the number of times a particular receiver location is used (i.e. inclusion in a patch) during the first round. In this case, it is one for all receivers since only one set of patches is extracted. Figure 5.2(b) shows the respective number for the second round.

Now, since the second set of patches is added, receivers from the second row up to the last 7 rows are included twice. This is because the second set starts from the second row (shifts down by one) and the last 7 rows are not included because they do not constitute

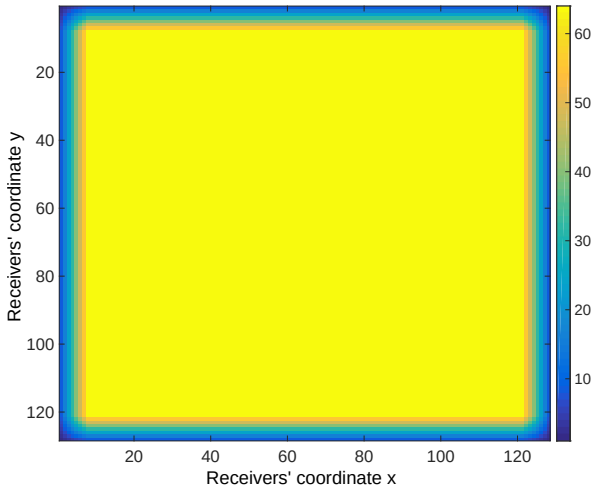


Fig. 5.3 Number of times each receiver is used in the section at the final Gibbs round. The receivers in the centre are used more frequently than those in the edges.

a complete 8×8 patch. After all 64 rounds, all sets are used at the same time to infer the variables of interest using $(128 - 7)^2 = 14641$ patches in total.

At the final iteration, each model variable is drawn from its corresponding distribution and used to calculate the receivers' value for all patches. Therefore, each receiver's value is inferred various times because it is contained in various patches (at most 64 in our example). Figure 5.3 shows the number of times each receiver is inferred when using 64 Gibbs rounds and having 1 Gibbs iteration per Gibbs round. It can be seen that the receivers at the edges of the section are contained in less rounds than the central receivers. In order to obtain the final values, the mean of each receiver's value is obtained by averaging over all its estimated values. In the same manner, the uncertainty of the prediction at that receiver location is obtained by calculating the variance of all its estimated values. Figure 5.4 shows an example of all estimated values for the receiver at location $(39, 39)$ which is estimated 64 times. Zhou et al. (2009) provide further information regarding the patch processing procedure. We will next discuss how to initialise all the variables in order to test the inference procedure. We will also give an interpretation of some of the variables.

5.3 BPFA variables and parameter settings

Figure 5.1 introduced all the model's variables and illustrated which parameters are necessary to be set. First, $\{c_0, d_0, e_0, f_0\}$ are parameters that describe the Gamma distributions. These are all set to 10^{-6} as is done usually to make them non-informative

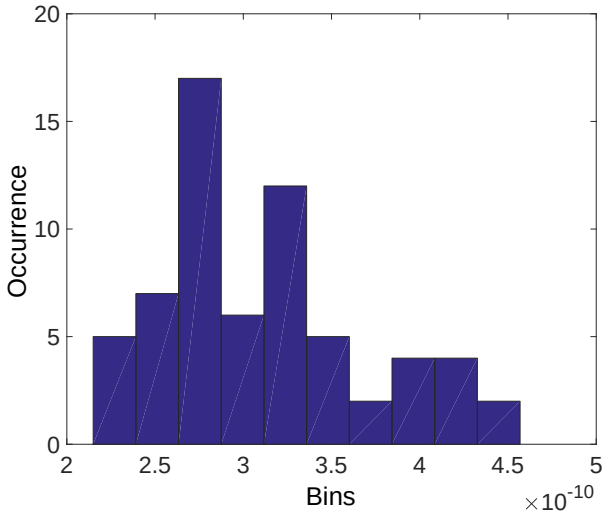


Fig. 5.4 Histogram of all 64 inferred values of a receiver at location $(39, 39)$. There are different values that this receiver can obtain and thus their mean is calculated for the final estimation. The variance can be used as part of an uncertainty map.

(Tipping, 2001). The parameters $\{a, b\}$ describe the Beta distribution that controls the probabilities as to whether a particular basis function generates a particular training subset. As discussed by Paisley and Carin (2009) as $L \to \infty$, the sparsity of $\mathbf{z}^{(i)}$ is described by a random variable drawn from a Poisson distribution, Poisson(a/b). Zhou et al. (2009) mention that the parameters $\{a, b\}$ are in general non-informative with the sparsity inferred from the data. Therefore, in practice we fix L to a specific number. We also set $a = 1$ and $b = T/8$ as specified in the BPFA software[1].

In our experiments, the upper limit of the dictionary's size, L, is set to $L = 256$. Similar results are obtained with $L = 512$ (Zhou et al., 2012) and therefore to reduce the computational cost, the former was set. However, in order to learn larger dictionaries of bases for further experimentation (for example in sections 5.7 and 5.9), we fixed this number and did not allow the algorithm to change its size. As discussed before, in equation 5.1, the training data can be extracted from many sections of time slices or from just one provided that there is enough data to prevent under-fitting. We performed the experiments on the reconstruction of 128×128 signals. Each signal was reconstructed individually, that is, for each 128×128 section, only training data from that signal were used. Each $\mathbf{x}^{(i)}$ is of size 8×8 extracted from the 128×128 section as discussed earlier.

[1]Zhou, M., 2012, Beta Process Factor Analysis software, http://mingyuanzhou.github.io/Code.html, accessed 4 May 2016

5.4 Initialisation, inference and analogy with POCS

All the unknown variables $\{\mathbf{z}^{(i)}\}_{i=1}^{T}, \{\mathbf{s}^{(i)}\}_{i=1}^{T}, \{\mathbf{d}_l\}_{l=1}^{L}, \{\pi_l\}_{l=1}^{L}, \{\boldsymbol{\epsilon}^{(i)}\}_{i=1}^{T}$ need to be inferred using the observed training data. Analytic equations for each variable are provided in section 5.1 and derived by Dang (2016) where the conditional probability distribution of each, conditioned on all others is obtained. Thus, it is possible to find an approximate solution by alternating between the variables, keeping the ones that have already been estimated fixed and estimating the one that is not fixed.

In order to start, all variables have to be initialised. $\mathbf{D}$ is initialised based on a Singular Value Decomposition (SVD) of $\mathbf{X}$ which converges faster as opposed to random initialisation or initialisation using other dictionaries (refer to section 5.9 for details). Furthermore, the noise precision, γ_ϵ, is initialised and scaled by the inverse variance of the available training data in a similar fashion as in Tipping and Faul (2003). This way, we ensure that the noise variance is not overestimated. Other variables are initialised randomly from their respective prior distributions.

An analogy can be drawn between BPFA and POCS. POCS transforms $\mathbf{X}$ to a pre-defined sparse domain (e.g. Fourier) and estimates the coefficients of the sparse transform of the data. The same idea of decomposing the data as the linear combination, $\mathbf{X} = \mathbf{DW}$, is used. $\mathbf{W}$ are the Fourier coefficients and $\mathbf{D}$ is the Fourier base where in the case of POCS the Fast Fourier Transform (FFT) is used for efficiency. One iteration of POCS is analogous to one iteration of BPFA for estimating the coefficients $\mathbf{W}$ but only partly. BPFA then considers the coefficients (or rather the variables that compose the coefficients $\{\mathbf{s}^{(i)}\}_{i=1}^{T}$ and $\{\mathbf{z}^{(i)}\}_{i=1}^{T}$) fixed and estimates a dictionary of bases, $\mathbf{D}$.

5.5 Lower limit for BPFA

Learning a dictionary of bases is not always possible since the training data must be sufficient. Not enough data can result in under fitting of the BPFA model by not adapting the bases to the available data. Therefore, it is useful to investigate the lower limit of the available percentage of training data for learning a dictionary of bases. Figure 5.5 shows an original section of a time slice that will be used for this experiment. We then remove receivers, using only 30% and 20% as seen in Figure 5.6(a) and Figure 5.6(b) respectively. Then, we use the BPFA to reconstruct the signal from 30% and 20% in Figure 5.6(c) and Figure 5.6(d) respectively. We can see that the reconstruction using 30% is significantly better as opposed to poor reconstruction using 20% of receivers. This is due to the fact that the dictionary of bases learned is different. Figure 5.6(e) shows the

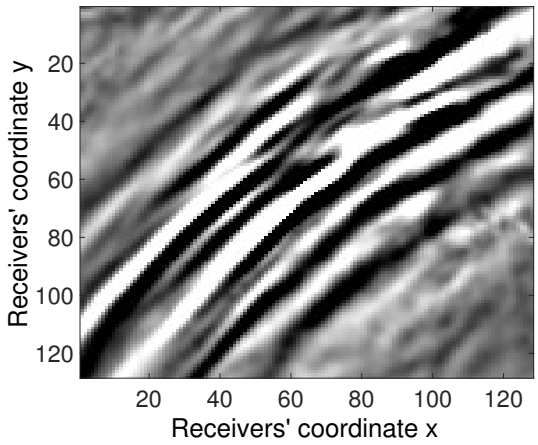
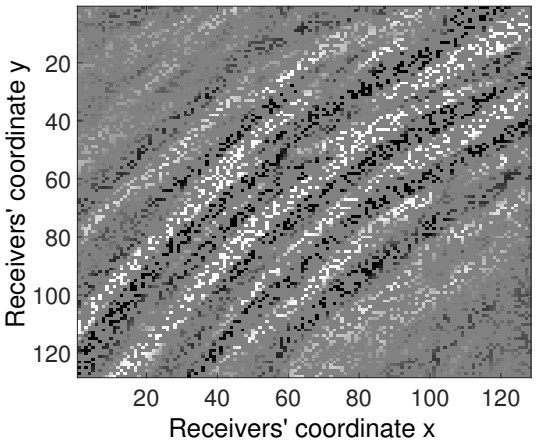


Fig. 5.5 Original section from a time slice to be used in experiment that investigates the lower limit of learning bases by the BPFA.

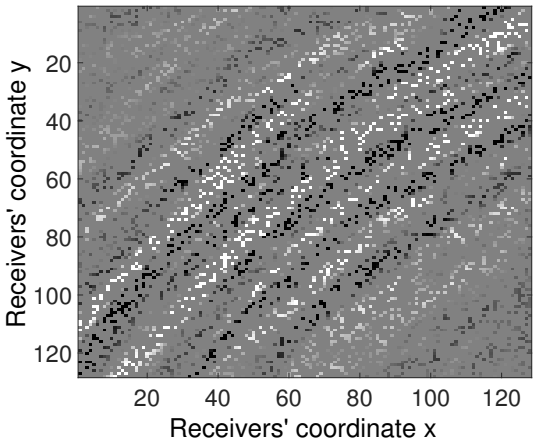
learned dictionary of bases from 30% which captures the largest variances in the data with success. On the other hand, Figure 5.6(f) does not show a dictionary of bases that captures any information resulting in a failed BPFA reconstruction. As we will see later on, with more sections of time slices, the model underfits with not enough training data and it already starts to perform badly with less than 30%.

Learning dictionary of bases per section

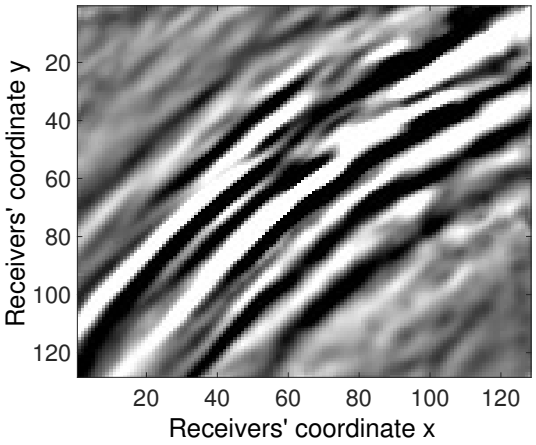
To avoid this lower limit per section, we could use patches from many sections and learn a dictionary using many signals. This might allow the learning of bases but it would provide a generic dictionary, not capturing the detailed characteristics per section. Given that each section contains seismic signals with different orientations and different variances, it is desirable to learn bases that are directly applicable to a section. In addition, in the field, the subsurface of the Earth varies from location to location and thus learning one universal dictionary on one type of subsurface image could be unusable in the next. This is similar to using one predefined dictionary of basis functions for all types of signal instances. We will show an example of a learned dictionary of bases which is generic and inferred from millions of signals in section 5.7 and compare it with dictionaries of bases learned per section used by various algorithms to show the advantages of the latter.



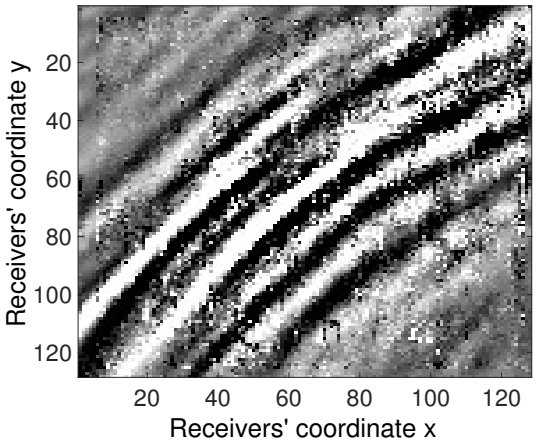
(a) Using 30% of receivers.



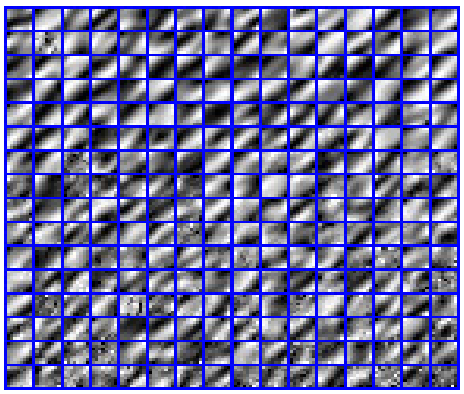
(b) Using 20% of receivers.



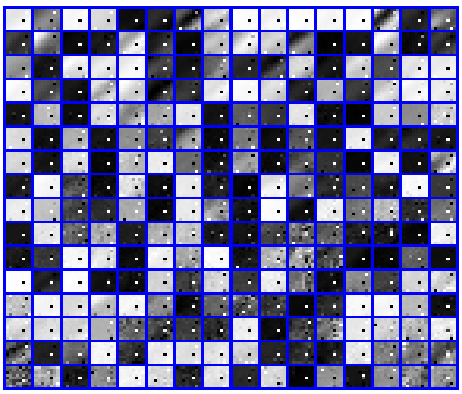
(c) BPFA recovery from 30%, Q = 28.382 db



(d) BPFA recovery from 20%, Q = 8.435 db



(e) Successfully learned bases from 30%



(f) Failed to learn bases from 20%

Fig. 5.6 We show the signal in (a) using 30% and in (b) using 20% of receivers. Also, we include in (c) the BPFA recovery for (a) and in (d) the BPFA recovery for (b). Lastly, (e) and (f) include the learned bases.

5.6 Reconstruction accuracy for time slices

In section 4.5, we first obtained the best configuration with regards to the reconstruction accuracy, Q, as defined in equation 4.32 for POCS, SPGL1 and the RVM by tuning their parameters. Then, we used the best option to perform comparisons on one hundred and fifty sections of time slices and on different percentages. As a reminder, 500 iterations are used for POCS for both 128×128 and 8×8 patch sizes. For SPGL1, patch sizes of 128×128 and 8×8 are used with the DCT bases. Finally, the RVM with DCT bases is used on both patch sizes and initialised with noise standard deviation, $\sigma_{no} = 10^{-11}$. We used both patch sizes even though the 128×128 configuration performs much better. This is due to the fact that we want to use the learned bases of BPFA (on 8×8 patches) with these algorithms and to compare the improvements gained by using learned as opposed to fixed bases.

Using these selected configurations, we use the same one hundred and fifty sections of time slices to compare against BPFA. BPFA parameters were fixed to the ones discussed in section 5.3 and 5.4. Figure 5.7 shows a reconstruction far from the source using the same signal as in chapter 4 in Figure 4.12 where the rest of the algorithms were compared. The original seismic section is given in Figure 5.7(a) and the same signal using only 50% of receivers is in Figure 5.7(b). The BPFA reconstruction is shown in Figure 5.7(c) obtaining better performance compared to SPGL1-DCT and POCS from Figure 4.12. Compared to the RVM-DCT, it obtains slightly worse reconstruction accuracy but the difference is very small. Figure 5.7(d) shows the dictionary of bases that was learned on this section using only the available training data from Figure 5.7(b). We can see that the orientation of the bases is similar to the original signal and captures the largest changes.

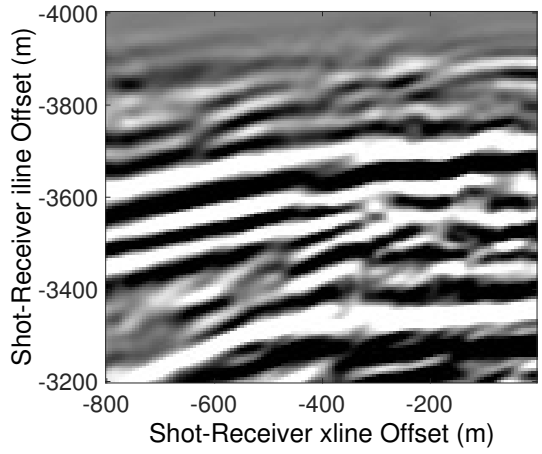
Another reconstruction, this time regarding a section closer to the source is included. Figure 5.8(a) shows the original section of this signal which was also used in Figure 4.13 for comparison with other algorithms. In this case, the BPFA reconstruction in Figure 5.8(c) obtains better reconstruction accuracy compared to all algorithms in Figure 4.13 even if it operates on 8×8 patches. Figure 5.8(d) shows the learned dictionary of bases using only 30% of receivers as seen in Figure 5.8(b). The dictionary is different than the one in Figure 5.7(d) and captures the general orientation of the signal.

In order to get a better understanding of performance over all one hundred and fifty sections, we plot the mean Q with different percentages of receivers used in Figure 5.9. We can see that the BPFA on 8×8 patches obtains very similar reconstruction accuracy as the RVM on 128×128 patch size, albeit slightly worse. Nevertheless, it is the best out of all 8×8 configurations of algorithms due to the fact that it learns dictionaries

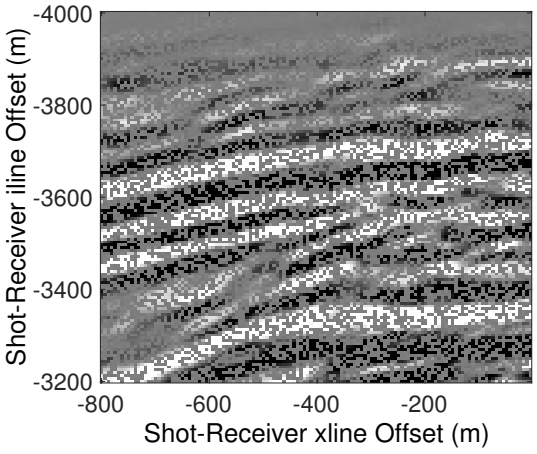
of bases dedicated to each section. It is also better than SPGL1-DCT and POCS even when they operate on 128×128 patches.

Collection of learned bases

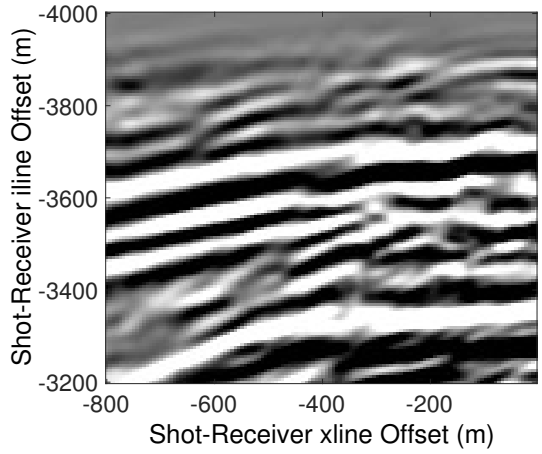
Using different sections of time slices with varying seismic signals, we obtain different dictionaries of bases. That is, for each section, a dictionary of bases is learned containing 256 bases. A collection of dictionaries of learned bases from different sections can be seen in Figure 5.10. We will see in the next section how we can use these to also improve the other algorithms along with example of reconstructions on the same seismic sections.



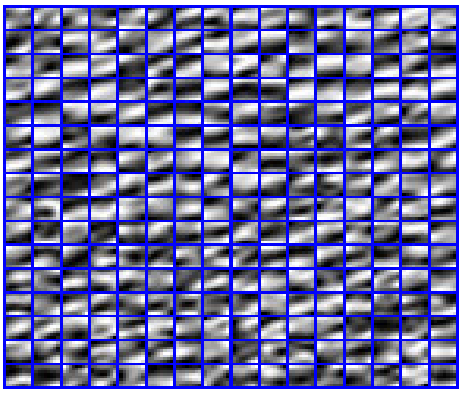
(a) Original



(b) 50% used

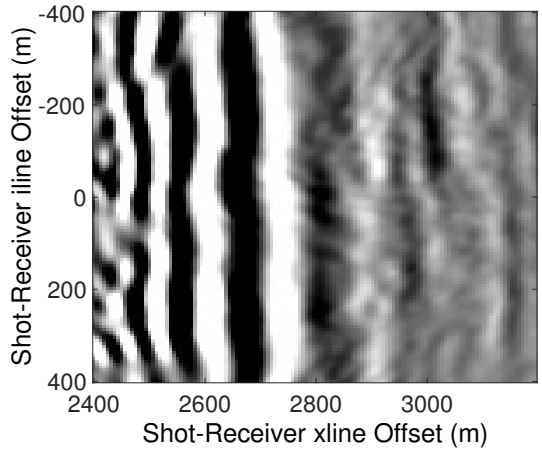


(c) BPFA, 8×8, Q = 41.20 db

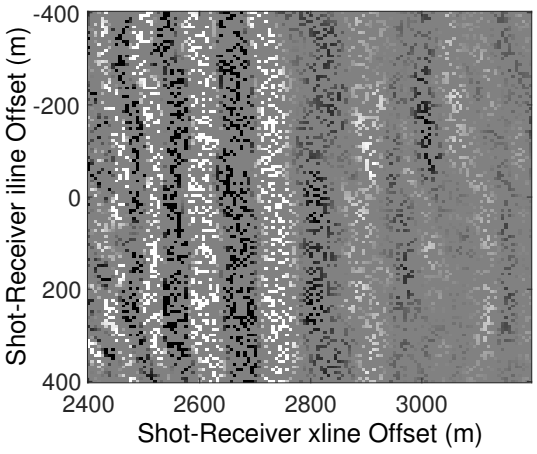


(d) BPFA bases

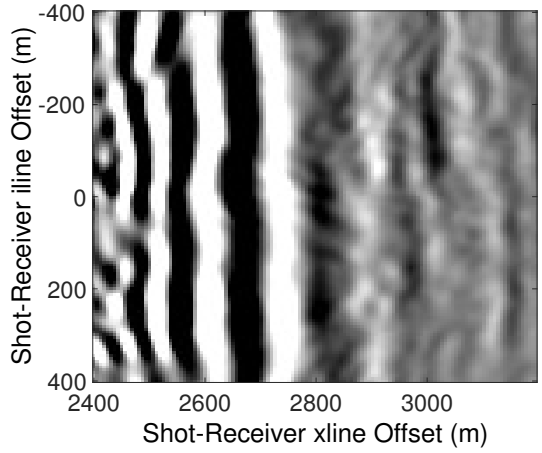
Fig. 5.7 A section from a time slice far from the source with original (a) and using 50% of receivers (b). BPFA reconstruction (c) and learned dictionary of bases (d) are included.



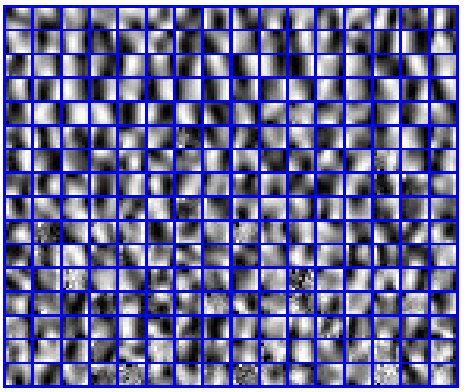
(a) Original



(b) 30% used



(c) BPFA, 8×8, Q = 38.52 db



(d) BPFA bases

Fig. 5.8 A section from a time slice closer to the source with original (a) and using 30% of receivers (b). BPFA reconstruction (c) and learned dictionary of bases (d) are included.

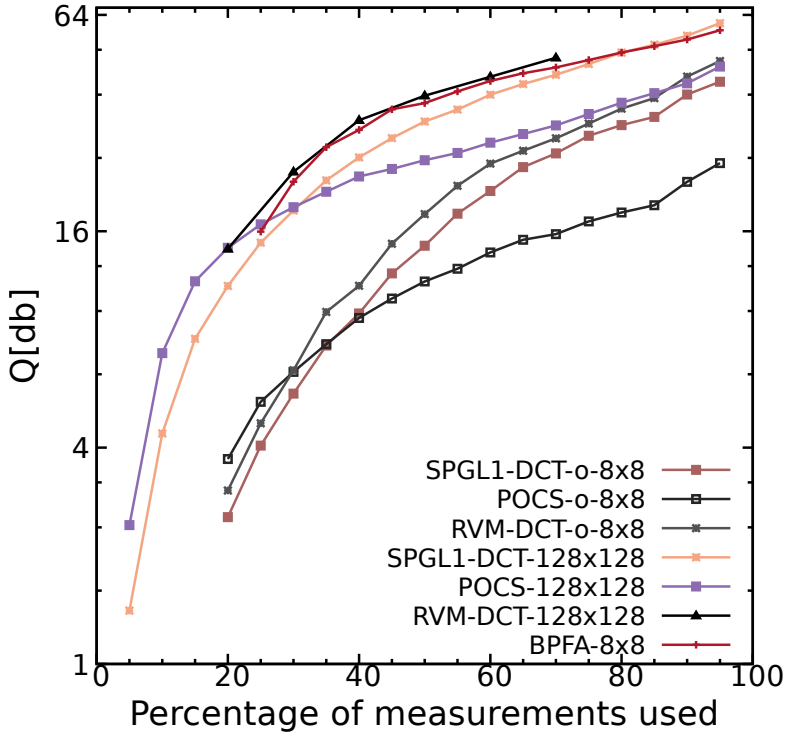


Fig. 5.9 Mean reconstruction accuracy over one hundred and fifty sections against the percentage of measurements used. BPFA obtains comparative accuracy with the RVM on 128×128 and is the best out of all 8×8 algorithms.

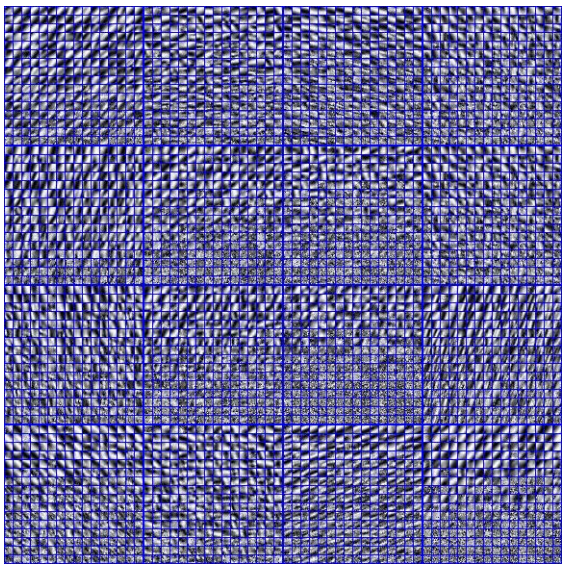
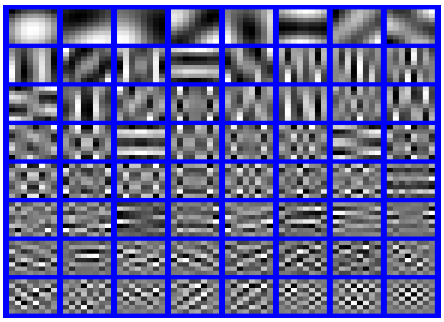
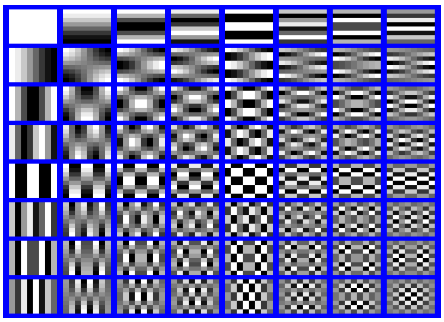


Fig. 5.10 Each dictionary of 256 bases is learned from an individual section of a time slice, resulting in as many dictionaries as reconstructions. An ensemble of dictionaries is available that captures different signal variations (depending on the time slice used for training) with different orientations of large changes.



(a) Inferred dictionary from 7 680 000 bases



(b) Discrete Cosine Transform

Fig. 5.11 Dictionaries of bases used in experiments.

5.7 Improving SPGL1 and RVM with learned bases

Using the Discrete Cosine Transform (DCT) with the RVM on 128×128 patches has provided the best mean reconstruction accuracy. Nevertheless, the computational time is very long and orders of magnitude slower than others as seen in Figures 4.15 - 4.17. Operating the RVM with the DCT bases on 8×8 patches reduced the computational time but simultaneously reduced the reconstruction accuracy. In addition, the assumption that the DCT provides a sparse representation for every section of a time slice is limiting.

In order to obtain a compromise between the two, it could help to operate on 8×8 patch sizes but using learned BPFA bases. We thus propose to use the BPFA bases that are learned for each section as a dictionary for the RVM and for the SPGL1. By doing this, we create two hybrid algorithms that use more specialised bases per section. To test their performance, we extracted 10 000 sections from the SEAM-II data set described earlier and ran the BPFA on three different percentages of receivers used (30%, 50% and 70%). Individual dictionaries for each section were learned resulting in 7 680 000 bases (10 000 with three percentages each providing a dictionary with 256 bases).

In addition to these, we wanted to obtain a universal dictionary of bases over all sections. To do this, we identify the signals' largest variations over all instances using Principal Component Analysis (PCA). This algorithm seeks an orthogonal projection of all the features onto a lower dimensional space. In particular, it seeks a domain where the variance of the projected data is maximised. We treated each basis as a feature and then used PCA to reduce this feature space to just 64 bases to match our input space by maximising the projected variance. Figure 5.11(a) shows this dictionary as opposed to the DCT shown in Figure 5.11(b). The learned dictionary is similar to the DCT having both high and low frequency components.

To evaluate the potential improvements of the learned dictionaries per section or the inferred bases from all sections, we ran experiments with the SPGL1 and the RVM using the DCT and the latter two dictionaries on the same 10 000 sections for all three percentages. For the learned dictionary of bases per section, we used the 64 most probable bases from the 256 bases. Table 5.1 illustrates the mean reconstruction accuracy in Q over all 10 000 sections. The first observation to make is that the BPFA on 8×8 patches is the best out of all other algorithms on the same patch size on all three percentages. Then, the RVM-DCT on 8×8 and the SPGL1-DCT obtain similar results with the former performing slightly better. When using the learned dictionary of bases from the BPFA per section, that is for each section the BPFA learns a dictionary and that dictionary is then used by the RVM and the SPGL1, the results are better. The RVM-Learned on 8×8 and the SPGL1-Learned on 8×8 perform significantly better than their DCT equivalents. This illustrates the importance of learning bases and we will see in chapter 6 if they provide alias free signals where we examine the behaviour in the x-t domain.

In addition, we investigated the effect of the inferred dictionary in Figure 5.11(a). We can see that using this, the RVM-All on 8×8 and the SPGL1-All on 8×8 do not provide big differences in reconstruction accuracy compared to the DCT. This is due to the fact that one global dictionary for sections with different signal characteristics is a big assumption both with our inferred dictionary and the DCT. In the case of seismic sections of time slices, one characteristic is the orientation of the edges of the waves.

We can group each section and each learned dictionary of bases in bins that characterise their orientation. We created 64 different dictionaries where each dictionary corresponds to two angles of the edge orientations ($0° - 180°$). Edge detection was first performed per section and then the histogram of the orientations of the edges was obtained. The dominant orientation was identified for that section and the dictionary of bases learned for that section was grouped with others in the relevant orientation bin. PCA was then performed in each group resulting in 64 dictionaries. Then, when the algorithms (the RVM and the SPGL1) reconstructed each section, its histogram of edge orientation was performed and the appropriate bin was chosen along with the corresponding dictionary of bases. The results are also included in Table 5.1. We can see that there is a small improvement in reconstruction accuracy as opposed to the inferred global dictionary and the DCT but not as great as the individual dictionary learned for that specific section.

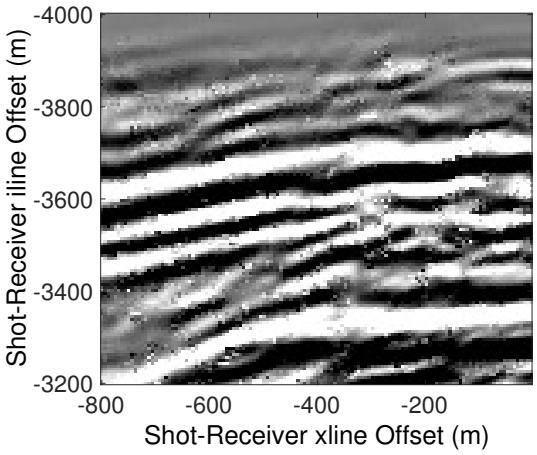
Therefore, we decided to continue the experiments and investigate further the effect on reconstruction accuracy of the individual learned dictionary of bases per section when used with the RVM and the SPGL1. Two examples of reconstructions of sections of time slices far and close to the source can be seen in Figures 5.12 and 5.13 respectively. The

10000 sections of time slices - Mean reconstruction accuracy in Q [db]			
Percentage used	30%	50%	70%
BPFA 8×8	**21.603**	**31.382**	**38.002**
SPGL1-DCT 8×8	9.007	17.466	26.118
SPGL1-All 8×8	10.434	18.986	26.650
SPGL1-Bins 8×8	11.488	20.156	28.025
SPGL1-Learned 8×8	19.561	28.126	34.971
RVM-DCT 8×8	10.854	19.326	27.748
RVM-All 8×8	10.000	18.365	26.264
RVM-Bins 8×8	10.939	19.446	27.434
RVM-Learned 8×8	17.917	26.248	32.195
POCS 8×8	9.524	14.408	18.940

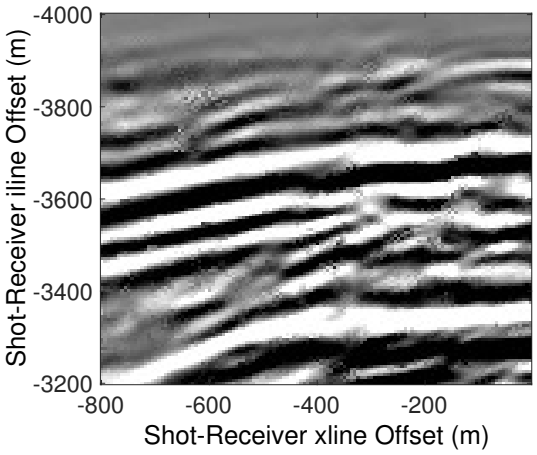
Table 5.1 Mean reconstruction accuracy from 10 000 sections of time slices using different configurations of algorithms and dictionaries. All (inferred dictionary using all other learned dictionaries by performing PCA), Learned (individual dictionary of bases learned by BPFA for a particular section) and Bins (categorising all individual dictionaries in 64 bins and then using PCA on each bin).

signals were also used when we compared the 128×128 versions of the algorithms and the BPFA on 8×8. These were given in Figures 4.12 and 5.7 for far from the source and Figures 4.13 and 5.8 for closer to the source.

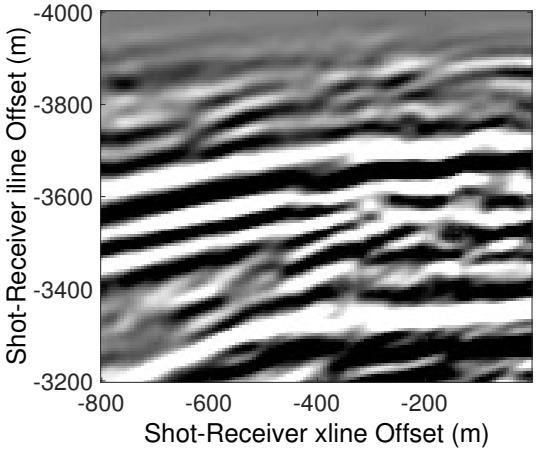
Figure 5.12(a) shows the POCS reconstruction on 8×8 with low Q illustrating that it requires larger patch sizes to work properly. The SPGL1-DCT on 8×8 performs better but not as good as its 128×128 version. The RVM-DCT on 8×8 performs even better but again not as good as its 128×128 version. When we change the dictionary of bases to the one learned by the BPFA, the performance of both the SPGL1 in Figure 5.12(c) and the RVM in Figure 5.12(e) improves dramatically and is close to the performance of their respective configurations on 128×128 albeit slightly worse. The same behaviour can be seen in Figure 5.13 with great improvements when using the learned bases from the BPFA. When the learned dictionary of bases is used by the SPGL1 and the RVM as seen in Figure 5.13(c) and Figure 5.13(e) respectively, the reconstruction accuracy greatly improves. This is also evident when we illustrate the reconstruction error maps. Figure 5.14 includes the reconstruction error maps for the reconstructions in Figure 5.13 and for the BPFA from Figure 5.8. It can be seen that there is small error when we use the BPFA and the hybrid algorithms using the learned dictionary of bases.



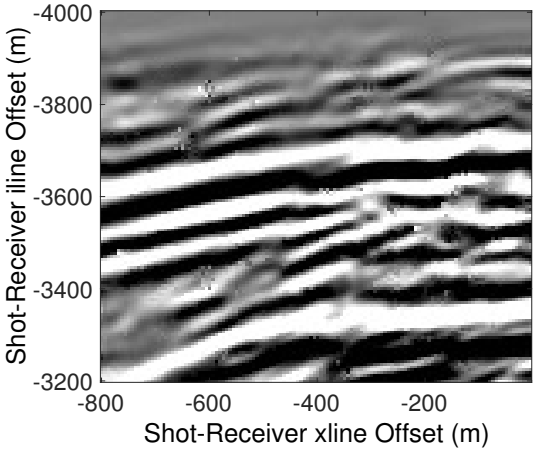
(a) POCS, 8×8, Q = 20.12 db



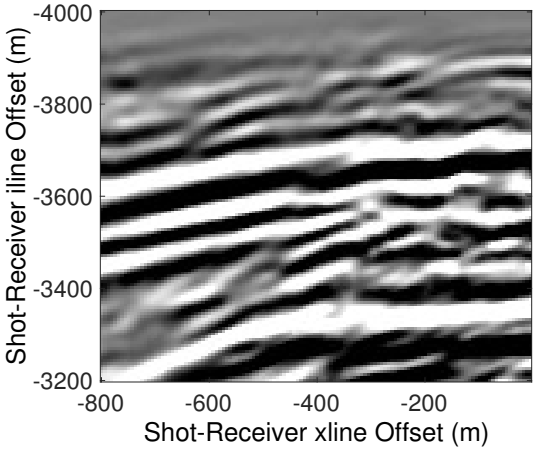
(b) SPGL1-DCT, 8×8, Q = 22.28 db



(c) SPGL1-learned, 8×8, Q = 35.40 db

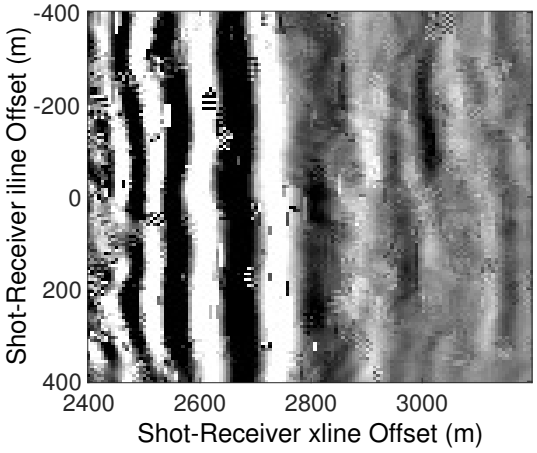


(d) RVM-DCT, 8×8, Q = 27.45 db

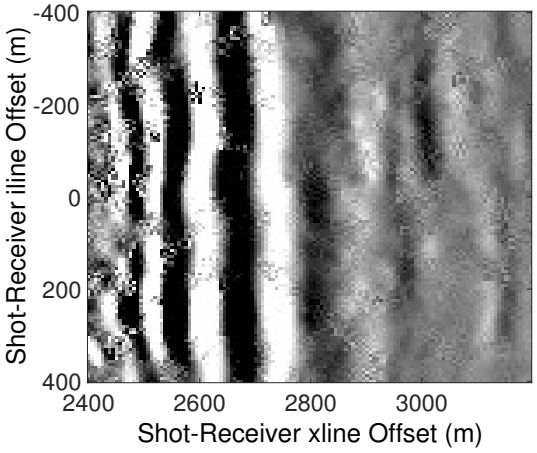


(e) RVM-learned, 8×8, Q = 38.70 db

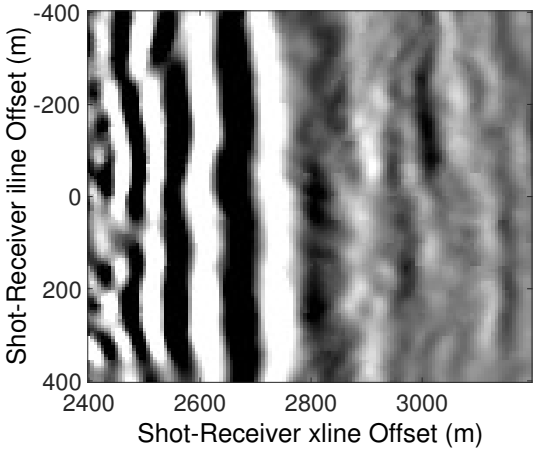
Fig. 5.12 Reconstructions for Figure 5.7(b) using 8×8 patches and Q as defined in equation 4.32. We show (a) POCS, (b) SPGL1 with DCT and (c) learned bases, (d) RVM with DCT and (e) learned bases.



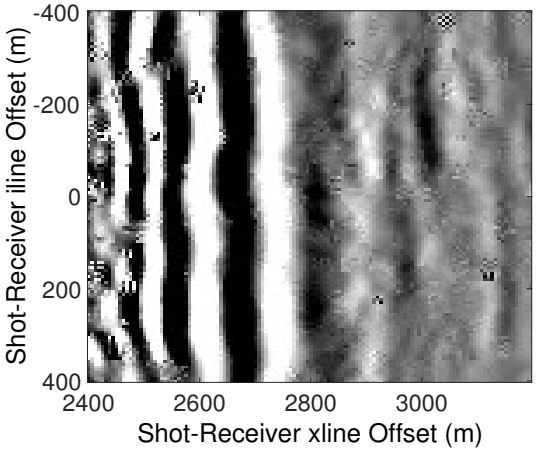
(a) POCS, 8×8, Q = 15.42 db



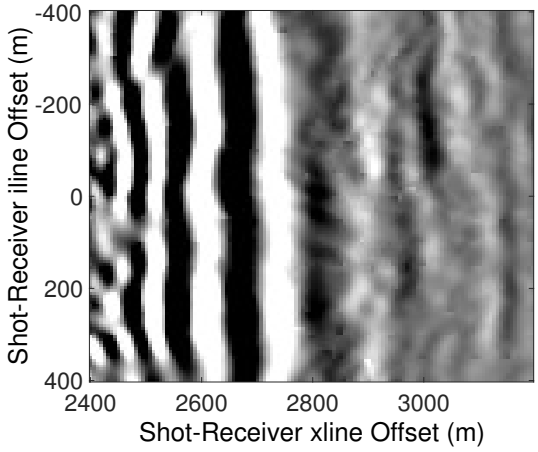
(b) SPGL1-DCT, 8×8, Q = 14.38 db



(c) SPGL1-learned, 8×8, Q = 31.87 db

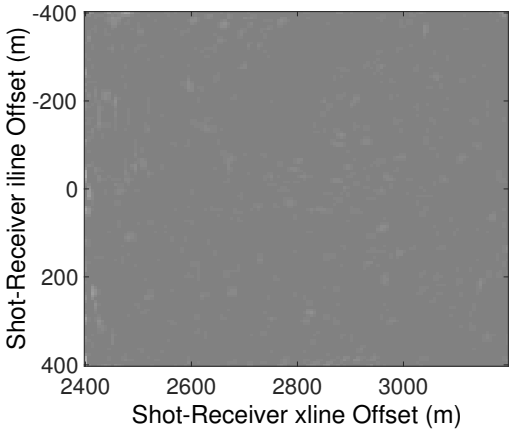


(d) RVM-DCT, 8×8, Q = 19.35 db

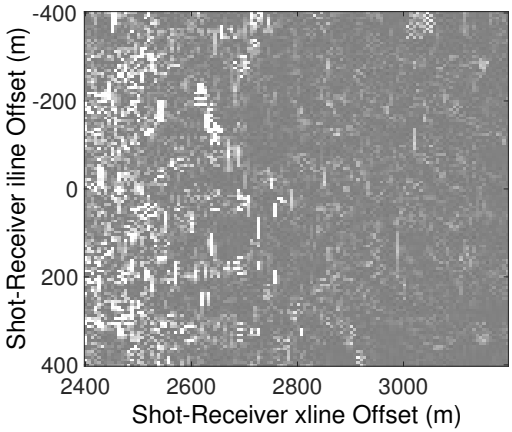


(e) RVM-learned, 8×8, Q = 32.14 db

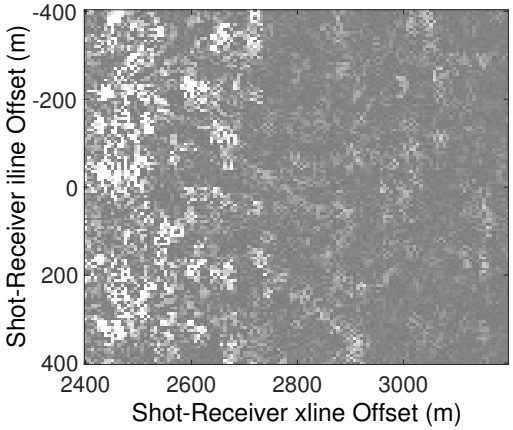
Fig. 5.13 Reconstructions for Figure 5.8(b) using 8×8 patches and Q as defined in equation 4.32. We show (a) POCS, (b) SPGL1 with DCT and (c) learned bases, (d) RVM with DCT and (e) learned bases.



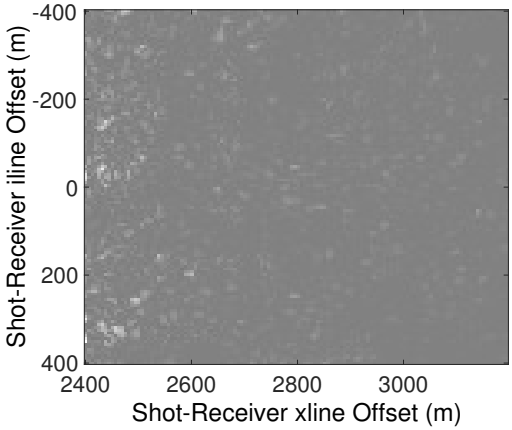
(a) BPFA, 8×8, error map



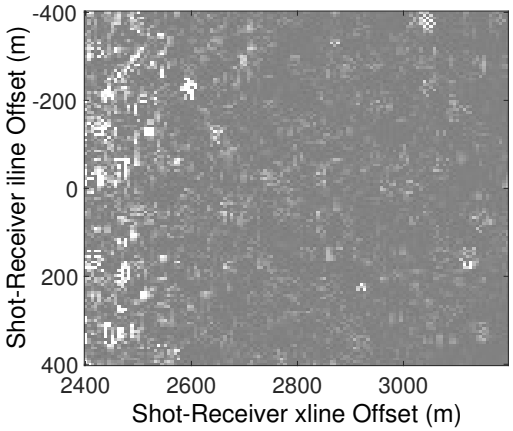
(b) POCS, 8×8, error map



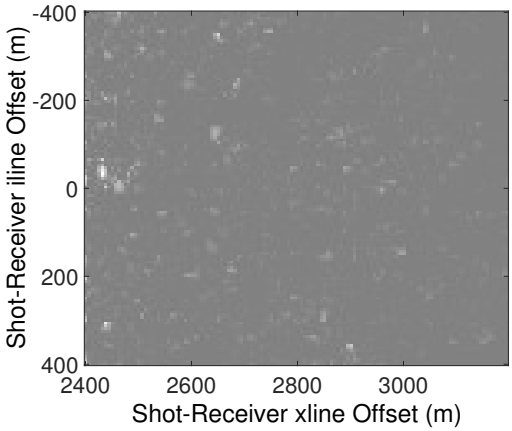
(c) SPGL1-DCT, 8×8, error map



(d) SPGL1-learned, 8×8, error map



(e) RVM-DCT, 8×8, error map



(f) RVM-learned, 8×8, error map

Fig. 5.14 We show the reconstruction error of (a) BPFA for Figure 5.8(c), (b) POCS for Figure 5.13(a), (c) SPGL1 with DCT for Figure 5.13(b), (d) SPGL1 with learned bases for Figure 5.13(c), (e) RVM with DCT for Figure 5.13(d), (f) RVM with learned bases for Figure 5.13(e). All algorithms use 8×8 patches.

5.8 Computational complexity and trade-offs

Obtaining high reconstruction accuracy is essential but is not the only criterion for the usage of an algorithm. Another criterion is its computational time. A discussion on computation was given in section 4.5 for POCS, SPGL1-DCT and RVM-DCT with computational times recorded and illustrated in Figures 4.15 - 4.17. In this section, we provide further details with regards to computation.

Depending on the convergence criteria, the algorithms could terminate earlier than expected, nevertheless the worst case scenario is mentioned. Spectral Projected Gradient for L1 (SPGL1) is composed of three potentially heavy computational steps, two matrix-vector products and a step that computes the projection of data. The worst-case complexity for the projection is $\mathcal{O}(N \log N)$ where N is the number of available receivers but on average it performs much better (van den Berg and Friedlander, 2009). Projection Onto Convex Sets (POCS) main computations are the Fast Fourier Transform (FFT) and Inverse Fast Fourier Transform (IFFT) which are $\mathcal{O}(N \log N)$ and is also dependent on the number of iterations until termination. Beta Process Factor Analysis (BPFA) scales linearly as a function of the patch size K, the dictionary size L, the sparsity level T_0 of the signals and the number of available training data T (Zhou et al., 2009). For the Relevance Vector Machine (RVM), we used the fast version which has $\mathcal{O}(Nb^2)$ with N the number of receivers and b the number of relevant bases chosen (Ji et al., 2008).

The orders of computational complexity are generally informative, however, since there are many algorithmic variations, we recorded the computational time to get a better understanding of their cost. Experiments were performed using the respective MATLAB packages mentioned in section 4.5 and for BPFA mentioned in section 5.3. All experiments were again performed as single-core jobs on machines with Intel(R) Xeon(R) CPU E5-2650 with 2.00GHz.

The mean computational time for three different percentages of receivers used (30%, 50% and 70%) and the reconstruction accuracy for all algorithms have been recorded over one hundred and fifty sections of time slices. Figure 5.15 shows the results when 30% of receivers are used. We can clearly see that the configurations of overlapping patches for POCS, RVM with DCT and SPGL1 with DCT on 8×8 patches obtain poor accuracy but are fast. The RVM and the SPGL1 with the learned bases from the BPFA for each section ran fast and at the same time obtain improved reconstruction accuracy. POCS and SPGL1 with DCT on 128×128 obtain better accuracy than the 8×8 configurations of the same algorithms. The BPFA on 8×8 obtains the best accuracy out of all the 8×8 configurations but it is also the slowest from those. The RVM with DCT on 128×128 obtains the best accuracy out of all but it is the slowest.

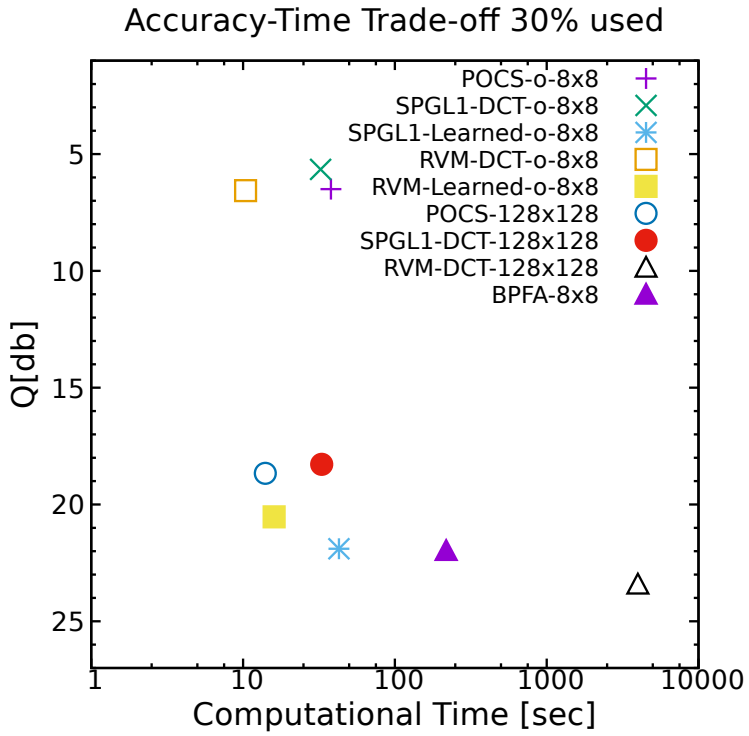


Fig. 5.15 Mean reconstruction accuracy, Q, against time using 30% of receivers.

When using 50% of receivers, similar behaviour in Figure 5.16 is obtained. It is worth noting now that the RVM-Learned on 8×8 patch size obtains better reconstruction accuracy than POCS and SPGL1-DCT on 128×128 and on 8×8 for all. Only BPFA and RVM-DCT on 128×128 are better than the RVM-Learned but the latter is orders of magnitude faster. Figure 5.17 exhibits similar behaviour when using 70% of receivers.

Overall, the best and slowest out of all algorithms is the RVM-DCT on 128×128 patches. Then, the BPFA is the best and slowest on 8×8 patches. The RVM-Learned on 8×8 is in general the second best out of the 8×8 configurations after BPFA. The difference in accuracy though is much smaller than the difference in speed. The RVM-Learned is orders of magnitude faster than both the BPFA and the RVM-DCT on 128×128 with Q only worse by a few db. The RVM-Learned provides the best compromise between the two. Nevertheless, the choice depends on the requirements of the survey. The RVM-Learned will still need to first learn the BPFA bases to use with the RVM (unless the bases were learnt from similar signals in the past). Therefore, speeding up the BPFA would be advantageous and we will examine this in the next section.

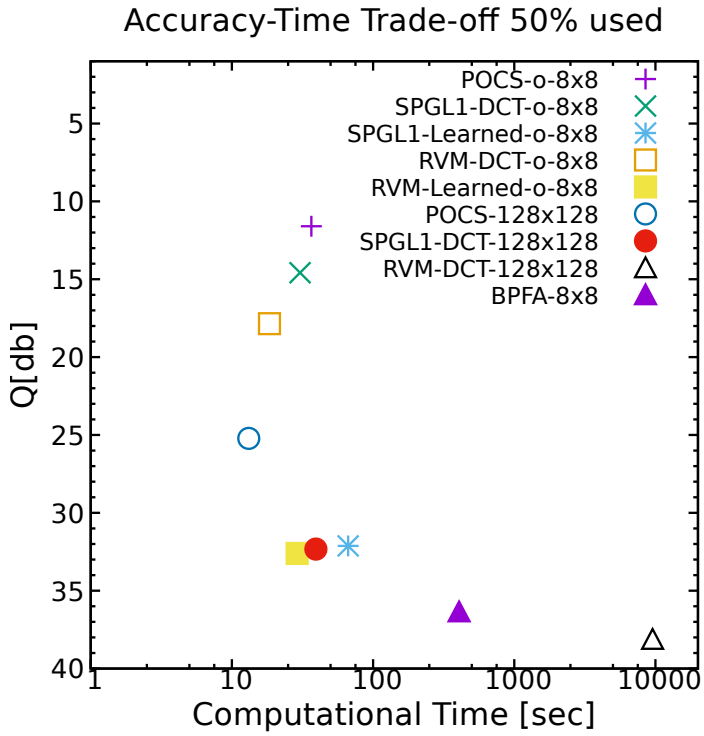


Fig. 5.16 Mean reconstruction accuracy, Q, against time using 50% of receivers.

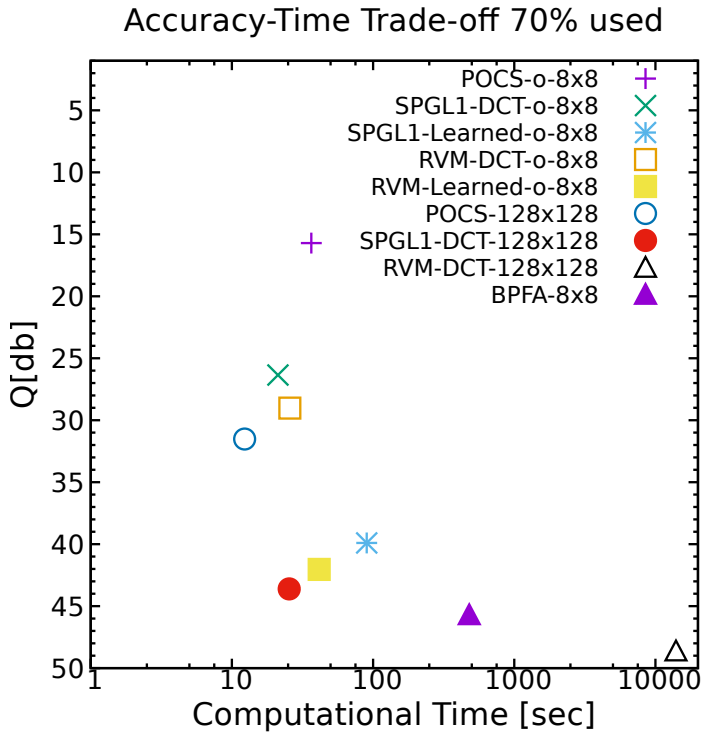
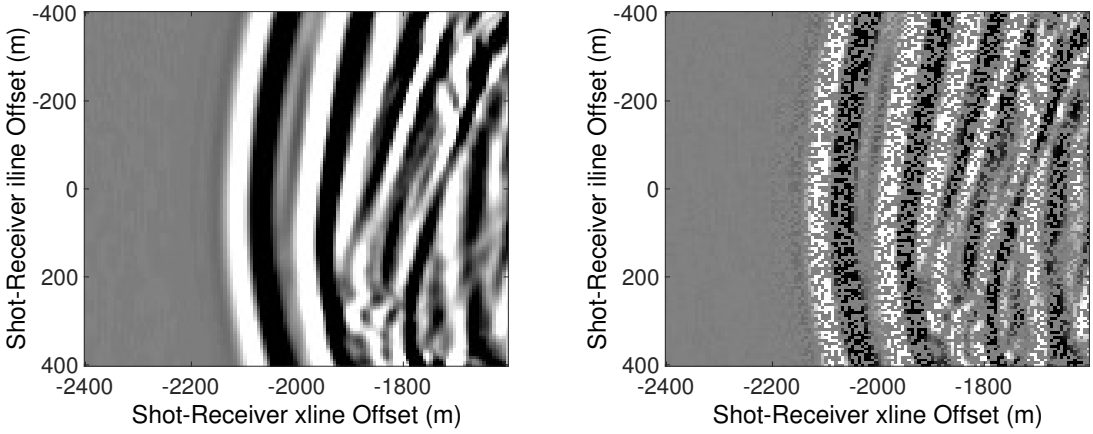


Fig. 5.17 Mean reconstruction accuracy, Q, against time using 70% of receivers.



(a) Original

(b) Using 50% of receivers

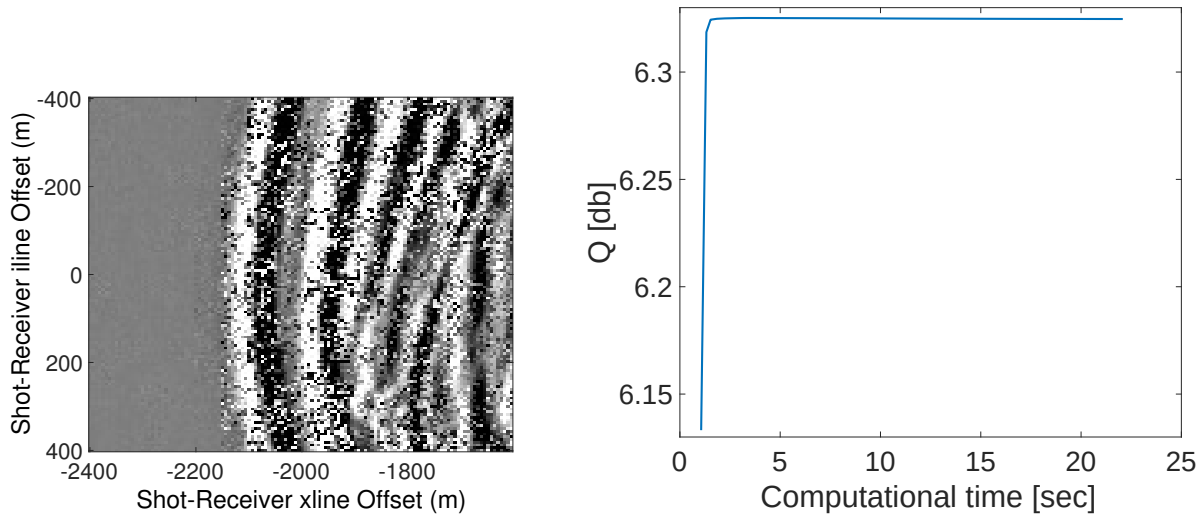
Fig. 5.18 Seismic section with (a) the original signal and (b) shows only 50% of the signal.

5.9 Gibbs analysis for faster BPFA inference

In the previous section, a comparison of BPFA against others has been provided with BPFA being orders of magnitude slower than the SPGL1, POCS and the RVM on 8×8 patch sizes. Considering that potentially thousands of instances of BPFA are executed for a complete seismic survey reconstruction, reducing the processing time can have great computational improvements.

One way to reduce computation is to reduce the amount of training data used. Instead of using patches with overlaps, we can extract directly the patches. In our case, we have 256 patches of 8×8. Figure 5.18(a) shows the original section of 128×128 receivers and Figure 5.18(b) shows 50% of receivers of the same signal. Using the BPFA with no overlaps, we obtain the signal in Figure 5.19(a) which has very low reconstruction accuracy. This is because there are not sufficient training data which results in under-fitting (learnt bases are not accurate). In this case, we used 100 Gibbs iterations with 256 patches (one round). Figure 5.19(b) shows how Q varies with time (number of iterations). In the first iteration, the variables are initialised randomly (except $\mathbf{D}$) and then revert to the appropriate values. Gradually the quality improves but by a very small amount which is deemed insignificant. Thus, we need to use more rounds with patch overlaps.

Before starting the inference for the Gibbs sampling, we initialise all unknown variables. By doing so, we place the sampler in a location at the variable's space. One of the most important variables in the BPFA model is $\mathbf{D}$ as defined in equation 5.1 and we investigate its initialisation. We will evaluate six different options. The first three are popular dictionaries of basis functions: the Discrete Cosine Transform (DCT), the Haar wavelets



(a) BPFA reconstruction with Q= 6.133db

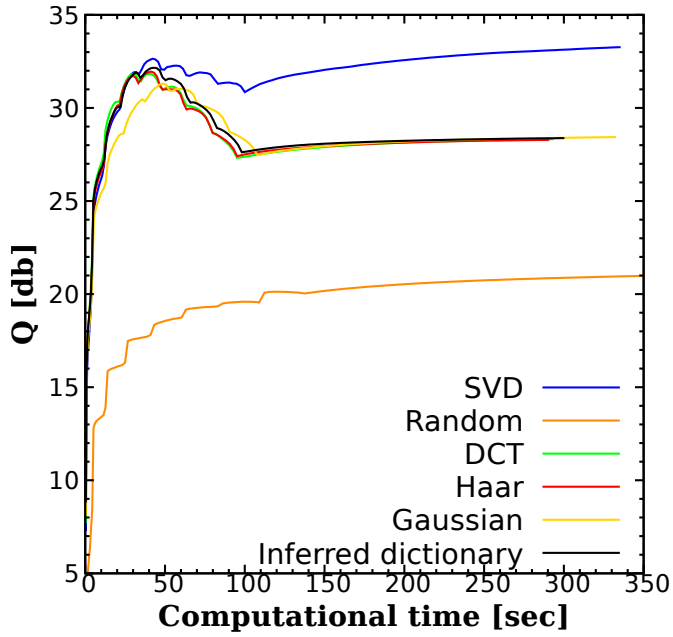
(b) Q vs time with no patch overlaps.

Fig. 5.19 BPFA reconstruction using no patch overlaps (a) and a plot of Q vs time (b).

transform and the radial (Gaussian) basis functions (definitions in section 2.4). Another option is the Singular Value Decomposition (SVD) which decomposes the available data into singular vectors that capture the largest variances in the data. A further option is the dictionary that we inferred by using 7 680 000 bases, from 30 000 seismic signals with 256 bases (refer to Figure 5.11). Random initialisation is also investigated.

Using these, we would like to investigate how the Gibbs samplers are affected. To do this, we use twenty different seismic sections of time slices and plot the mean Q at each time for each different initialisation of bases. Figure 5.20 shows the mean Q against the mean computational time. All Gibbs samplers exhibit similar behaviour. At the beginning of the inference, Q rapidly increases with the extraction of patches. Every 8 Gibbs rounds, there is a rapid change in Q when a horizontal shift in the extraction of patches occurs. This increases Q at the start of the inference but decreases it afterwards. When 63 Gibbs rounds are completed (with one iteration per round), the 64th begins using 100 iterations with all available patches as training data. This results in a better estimation and Q gradually increases. From the various initialisations, the Gibbs sampler with SVD performs much better. The inferred dictionary from 30000 sections peaks near the former's performance. The three dictionaries of basis functions have similar performance, but the Gaussian basis functions peak slightly lower. Finally, the random initialisation performs the worst.

Figure 5.21 illustrates the BPFA reconstruction at various instances in the Gibbs sampling with SVD initialisation. Alongside the signal, the reconstruction accuracy,

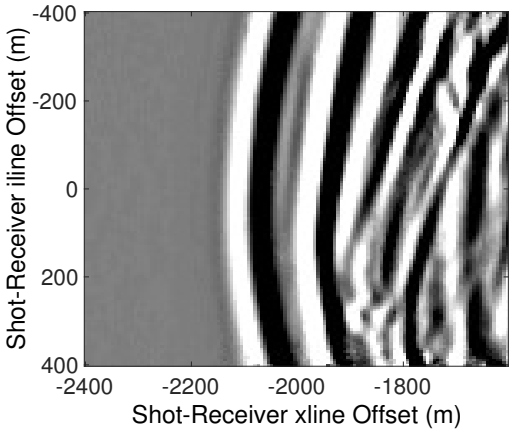


(a)

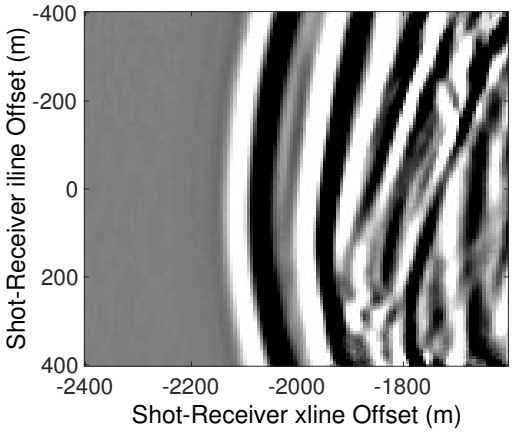
Fig. 5.20 Mean reconstruction accuracy, Q, of twenty sections against computational time.

Q, and the computational time are included. Figure 5.21(a) shows that already at 6.41 seconds, the quality reaches a high value. Then, as more Gibbs rounds occur, the quality increases more with not significant differences. Figure 5.22 illustrates the BPFA reconstructions after all rounds are used except the last one which continues with 100 iterations. The differences in reconstruction accuracy in the final round are minimal, with small increases in Q but large increase in computational time. In the last round, the difference in accuracy is approximately only 1.5 db but the computational time increases by 227.53 seconds. By using the insight from this analysis, we can speed up the BPFA. From Figure 5.20, we can see that Q does not improve significantly at the last Gibbs round. Thus, not all 100 iterations are necessary.

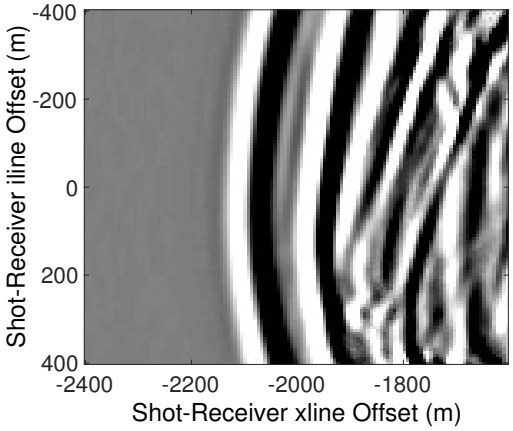
For the experiments in chapter 6, with slice processing and re-sorting in the x-t domain, we propose to use all 64 Gibbs rounds to reach high Q but stop the iterations at 50 with SVD as initialisation since it obtains the best Q. This allows a speed up of approximately 120 seconds (or 2 minutes) per section and only approximately 0.5 db loss. We are interested in the reconstruction of entire seismic signals which are composed of numerous sections. In particular, in our experiment we use 10000 sections of time slices (10 sections per time slice with 500 time steps far and near the source) which results in 20000 minutes or 333.3 hours of speed up.



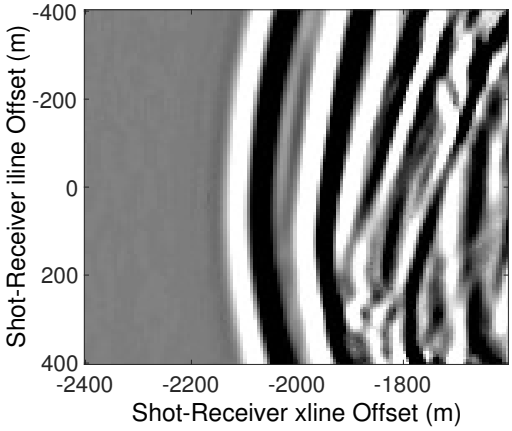
(a) R= 11, Q= 24.424 db, t= 6.41 secs



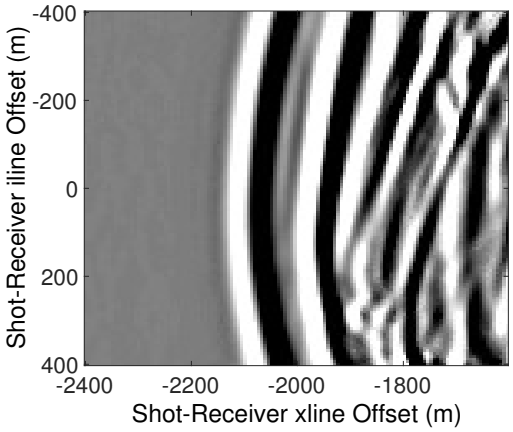
(b) R= 21, Q= 29.755 db, t= 17.03 secs



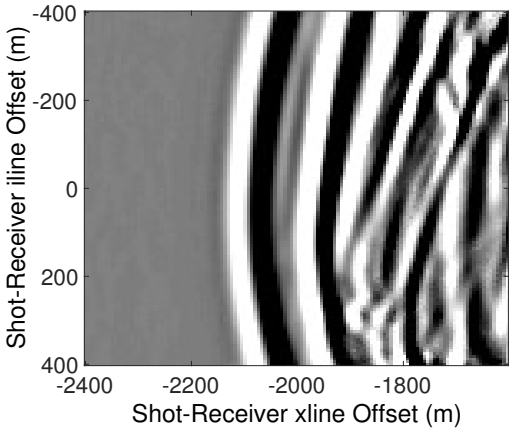
(c) R= 41, Q= 32.617 db, t= 48.39 secs



(d) R= 51, Q= 31.761 db, t= 67.91 secs

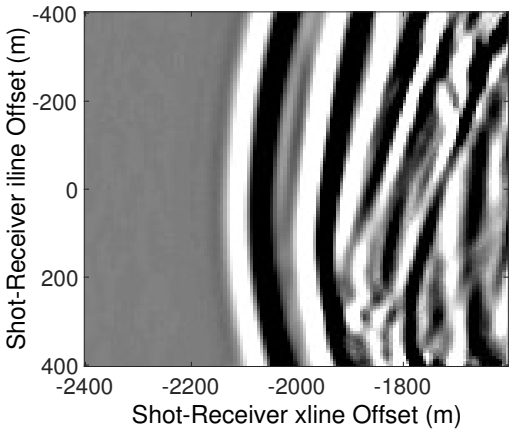


(e) R= 61, Q= 30.263 db, t= 89.82 secs

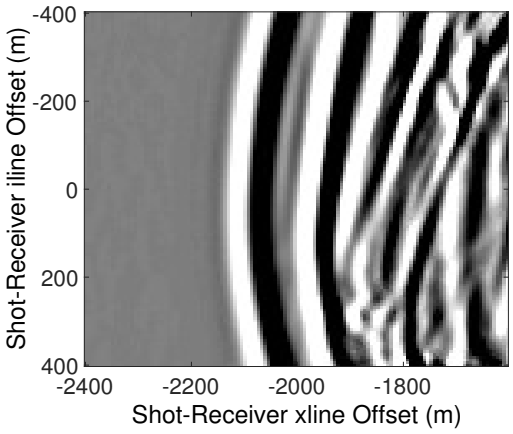


(f) R= 64 (1), Q= 29.903 db, t= 96.96 secs

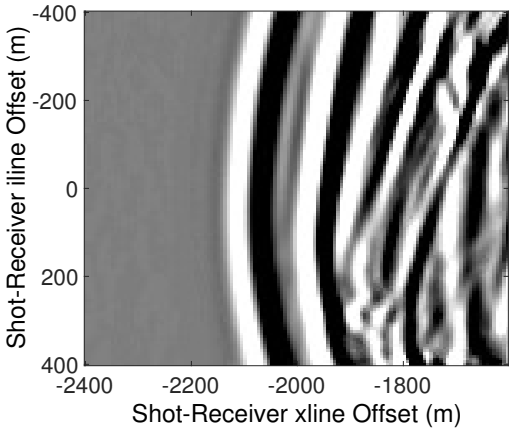
Fig. 5.21 BPFA reconstructions at various instances in the Gibbs sampling. R stands for rounds and t stands for computational time. In brackets, the iteration in the corresponding Gibbs round. We use Q as defined in equation 4.32.



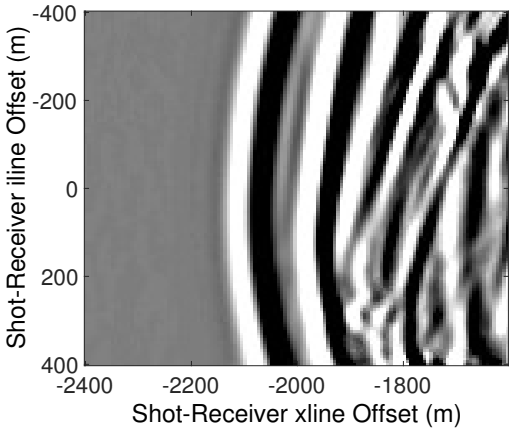
(a) R= 64 (25), Q= 30.432 db, t= 151.99 secs



(b) R= 64 (50), Q= 30.993 db, t= 210.23 secs

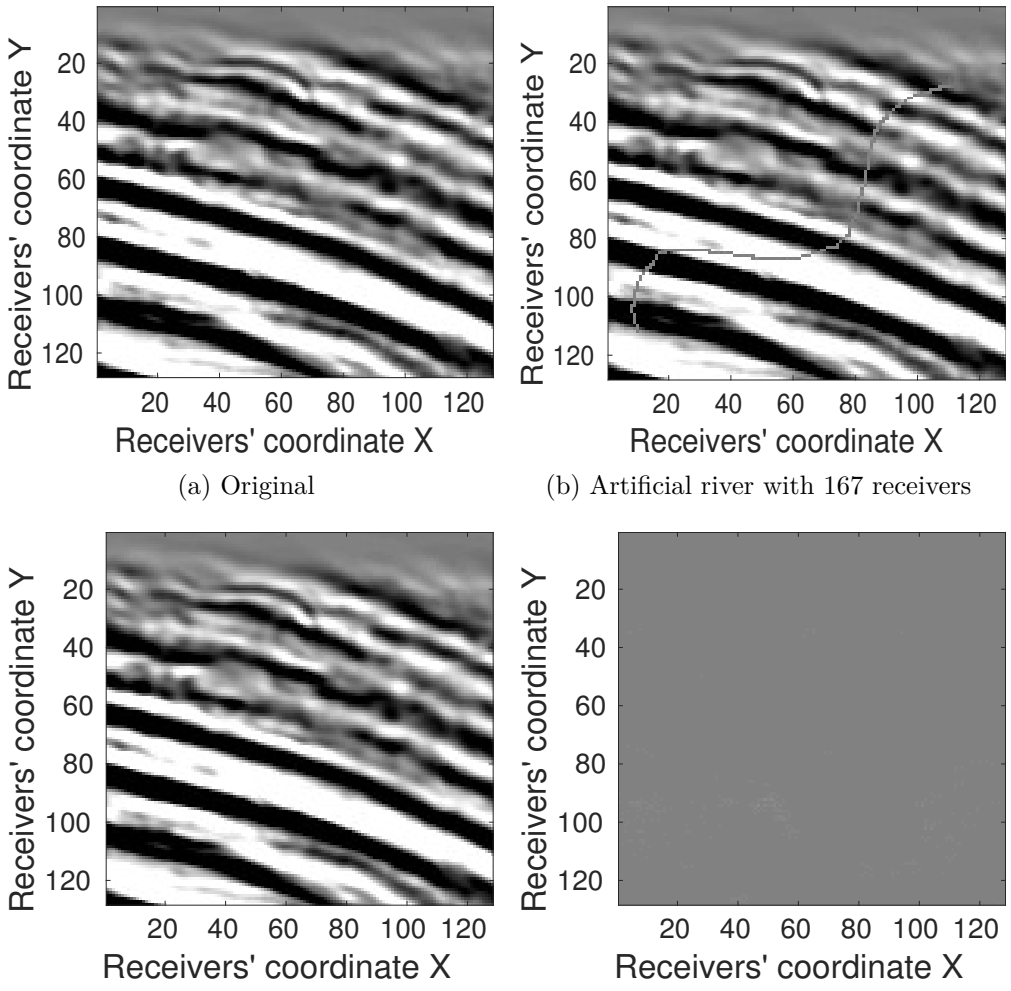


(c) R= 64 (75), Q= 31.292 db, t= 271.26 secs



(d) R= 64 (100), Q = 31.530 db, t= 330.59 secs

Fig. 5.22 BPFA reconstructions during last Gibbs round. R stands for rounds and t stands for computational time. In brackets, the iteration in the corresponding Gibbs round. We use Q as defined in equation 4.32.



(a) Original

(b) Artificial river with 167 receivers

(c) BPFA Reconstruction

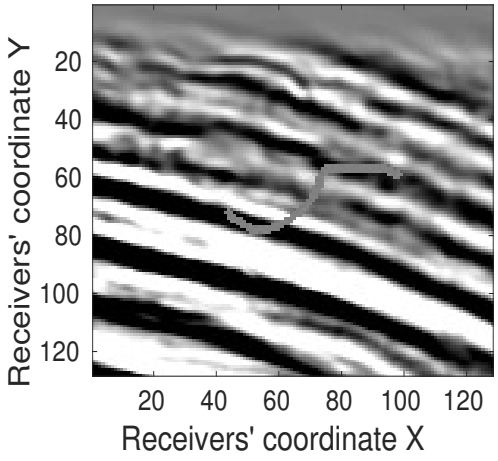
(d) Reconstruction error

Fig. 5.23 Reconstruction of an artificial river spanning 167 receivers using BPFA.

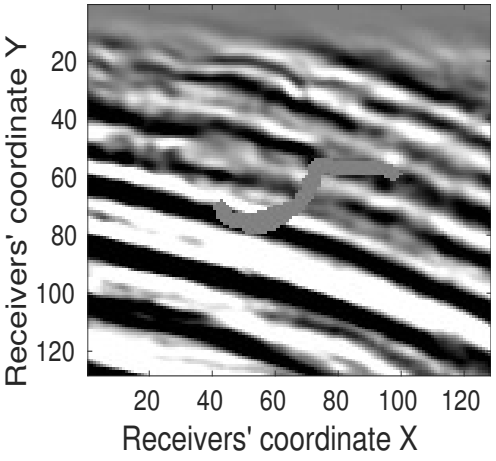
5.10 Artificial rivers and missing blocks

In seismic surveys, surface obstructions (e.g. rivers, buildings, etc.) preclude the placement of receivers. However, signal reconstruction algorithms should be able to recover as much of the signal as possible in such locations. To illustrate the behaviour of BPFA in these scenarios, we created various artificial rivers with varying lengths and widths.

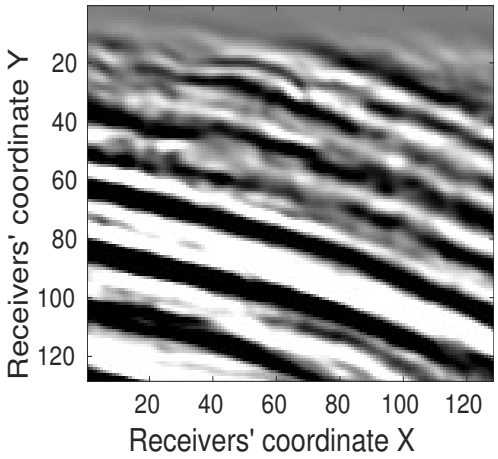
Figure 5.23(b) illustrates an artificial river spanning 167 receivers and the respective BPFA reconstruction can be seen in Figure 5.23(c). Two other examples can be seen in Figure 5.24 where the artificial river spans 193 receivers in Figure 5.24(a) and 390 receivers in Figure 5.24(b).



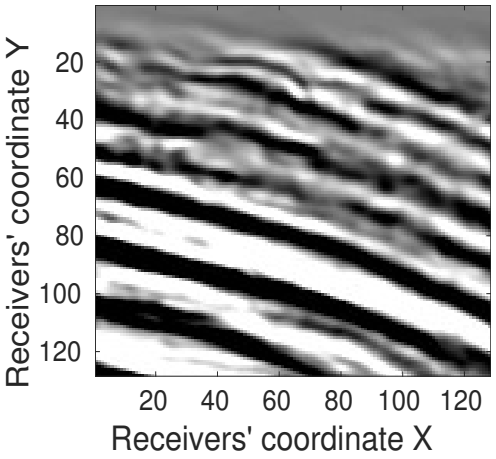
(a) Missing 193 receivers



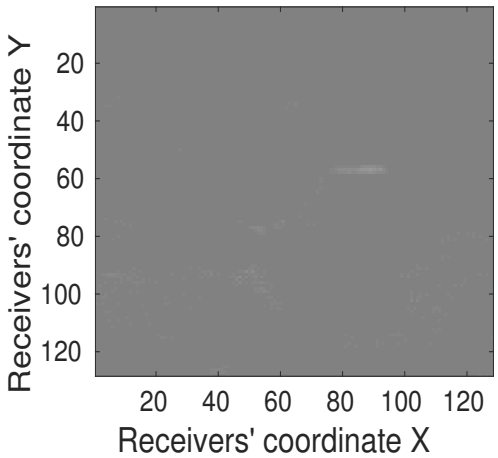
(b) Missing 390 receivers



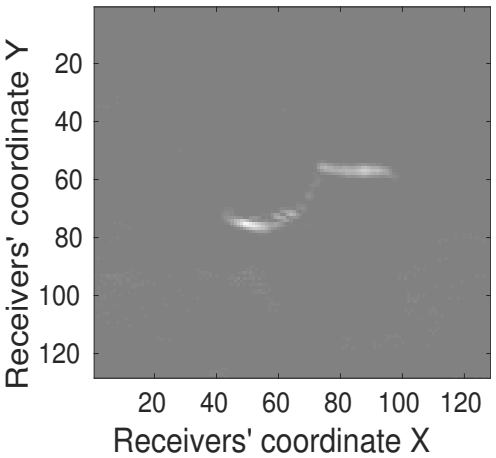
(c) BPFA reconstruction with 193 missing



(d) BPFA reconstruction with 390 missing

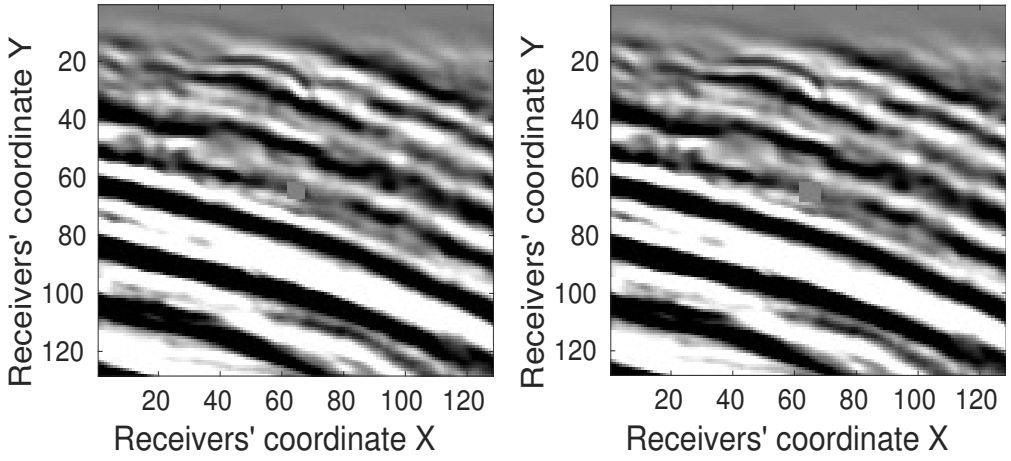


(e) Reconstruction error with 193 missing



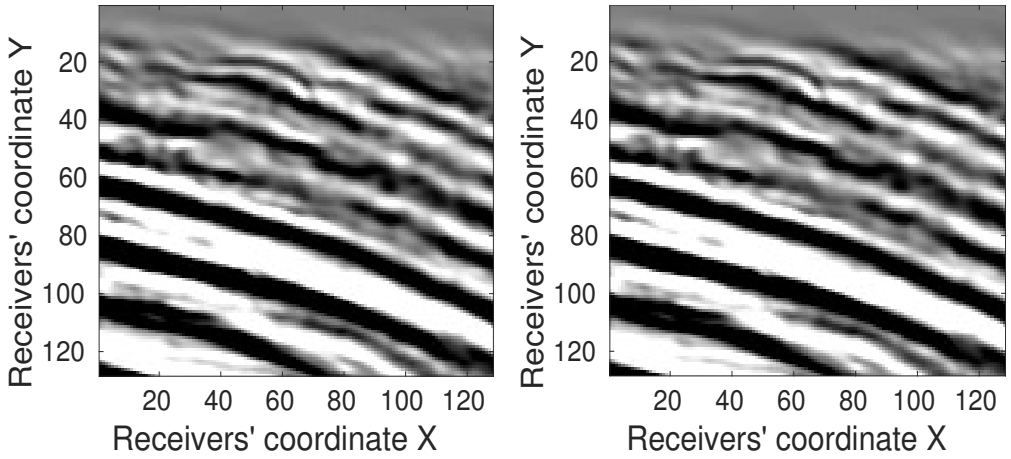
(f) Reconstruction error with 390 missing

Fig. 5.24 Reconstructions of artificial rivers spanning 193 and 390 receivers using BPFA.



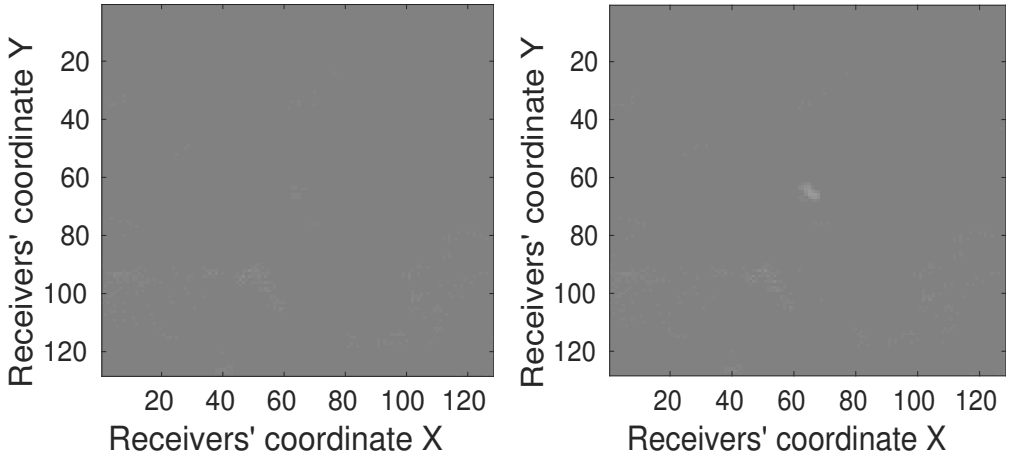
(a) Missing 6×6 receivers

(b) Missing 7×7 receivers



(c) BPFA Reconstruction with 6×6 missing

(d) BPFA Reconstruction with 7×7 missing



(e) Reconstruction error with 6×6 missing

(f) Reconstruction error with 7×7 missing

Fig. 5.25 Reconstruction of missing block of 6×6 and 7×7 receivers using BPFA.

BPFA is able to reconstruct the signal albeit with some reconstruction error as seen in Figures 5.24(e) and 5.24(f) respectively. The larger the artificial river the more difficult it is to reconstruct it, as expected.

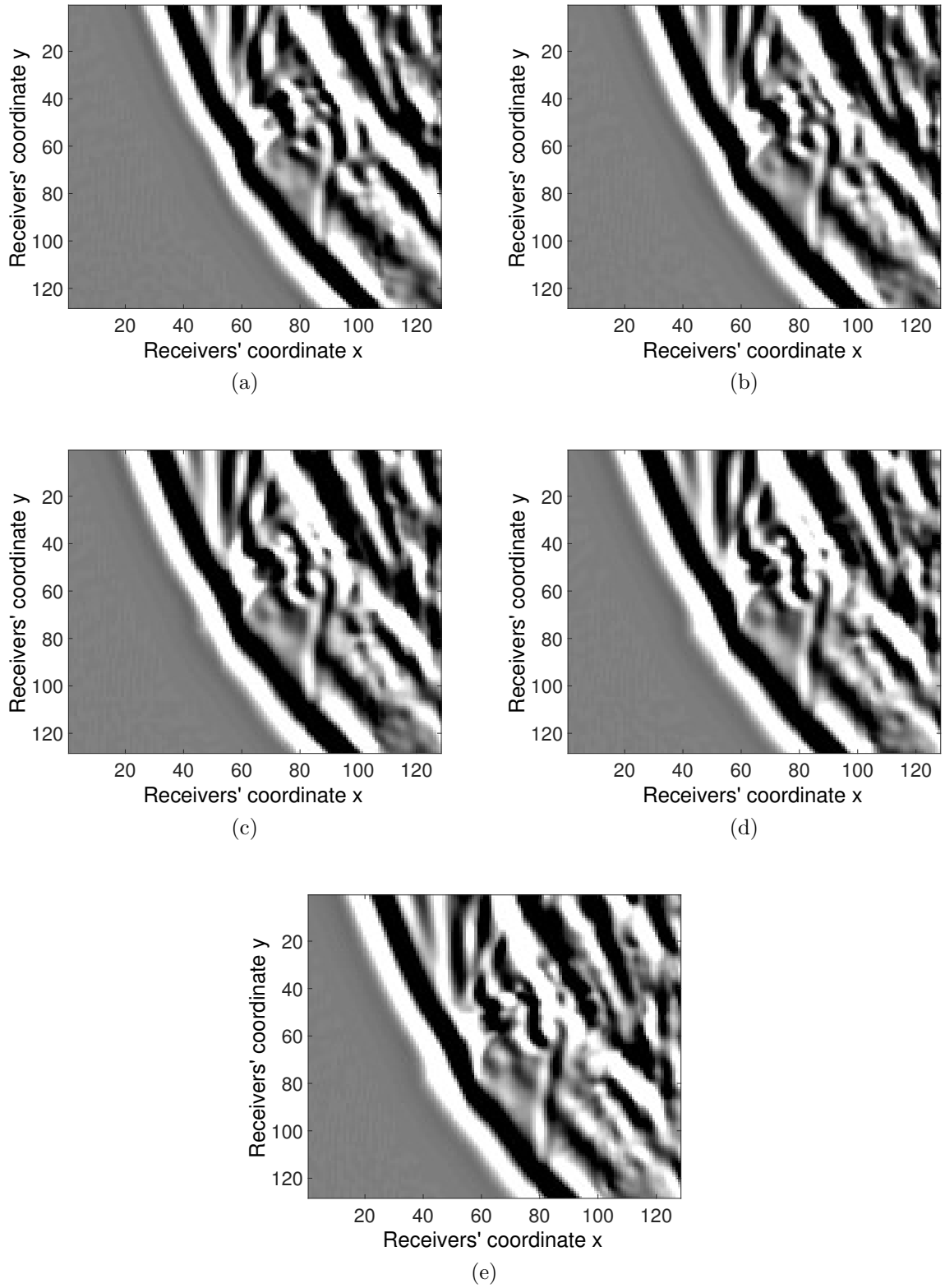
Another scenario that we wanted to test is whether it is possible to reconstruct missing blocks of receivers. Figure 5.25 show two examples of missing blocks of size 6×6 in Figure 5.25(a) and 7×7 in Figure 5.25(b). BPFA reconstructs the signal almost perfectly in both cases. Larger gaps with more consecutive receivers missing are not possible with the current operational patch size of 8×8 in 2D. In the next section, we will discuss the possibility of a three-dimensional BPFA implementation.

5.11 3D BPFA

BPFA is able to predict missing receivers' values using the available data from time slices very effectively. The input space that the algorithm has been operating so far is a 2D 8×8 space inside a 128×128 section. Nevertheless, seismic data are usually obtained in 3D volumes and thus we will investigate whether it is possible to apply it directly on a cube of $128 \times 128 \times 128$. The 3D algorithm acts on a $8 \times 8 \times 8$ space.

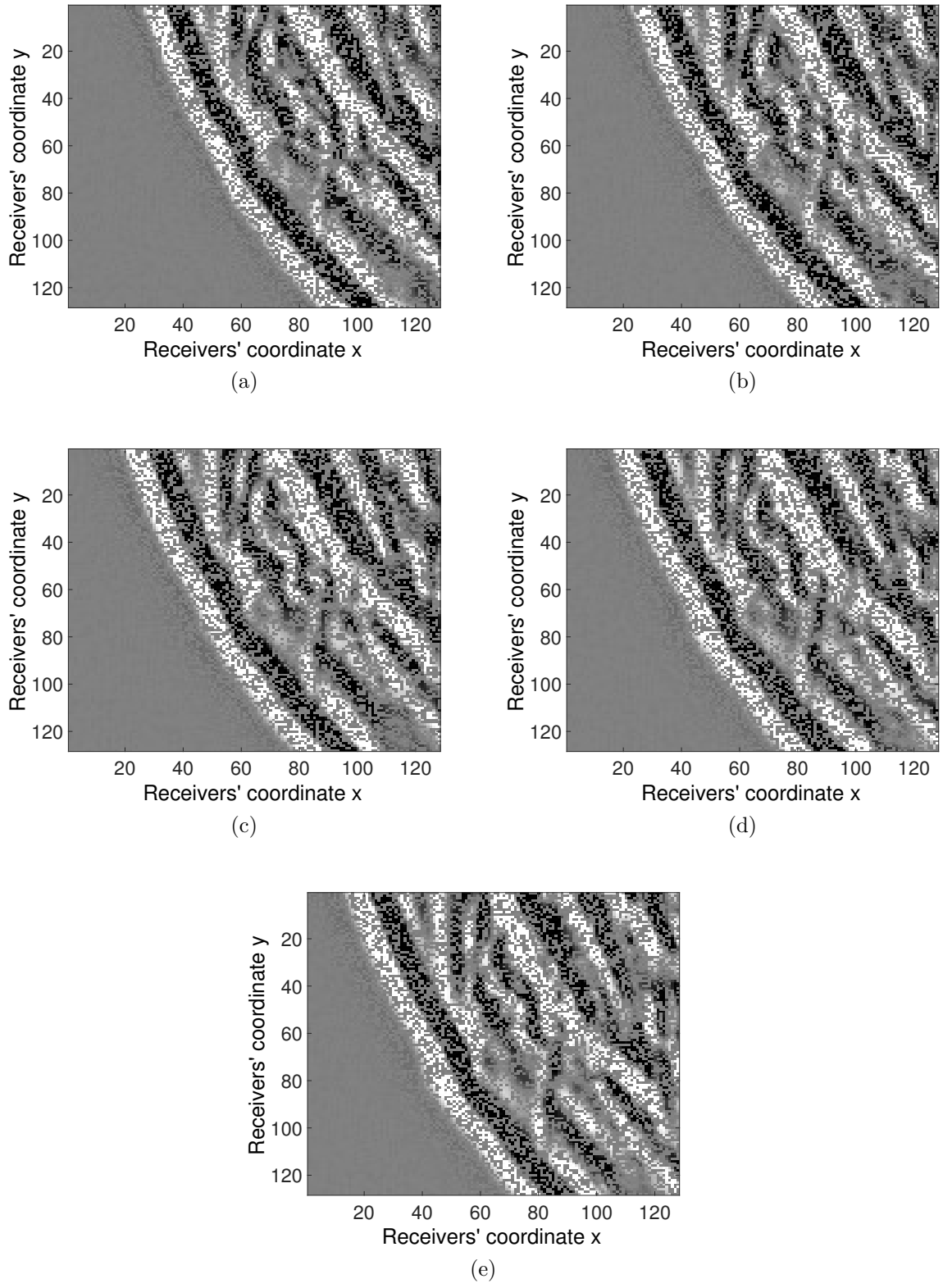
Figure 5.26 shows five sections from the original $128 \times 128 \times 128$ cube extracted from the SEAM-II data set. We then randomly removed 50% receivers from these with a fixed mask per time step resulting in the sections in Figure 5.27. Figure 5.29 shows sections from the 3D bases dictionary learned by BPFA. It can be seen (for example the third or fourth basis at the top row from left to right) that some bases evolve in time and are coherent illustrating that the algorithm is learning meaningful 3D bases. Nevertheless, some bases contain noise and are incoherent. The BPFA reconstruction can be seen in Figure 5.28 where sections from the 3D reconstruction are provided for illustration purposes. From these reconstructions, it can be seen that the algorithm did not converge and the result is poor.

We set the iterations to 50 which resulted in a running time of 130720 seconds or approximately 36 hours. The introduction of another dimension increased the running time substantially from approximately 210 seconds in 2D. Due to the increase in dimensions, the iterations should also increase but we deemed that this would be impractical with the current implementation. We will show in chapter 6 that is is possible to reconstruct 3D volumes indirectly by reconstructing 2D signals (time slices) in time resulting in a pseudo 3D reconstruction with great reconstruction accuracy and no increase of iterations.



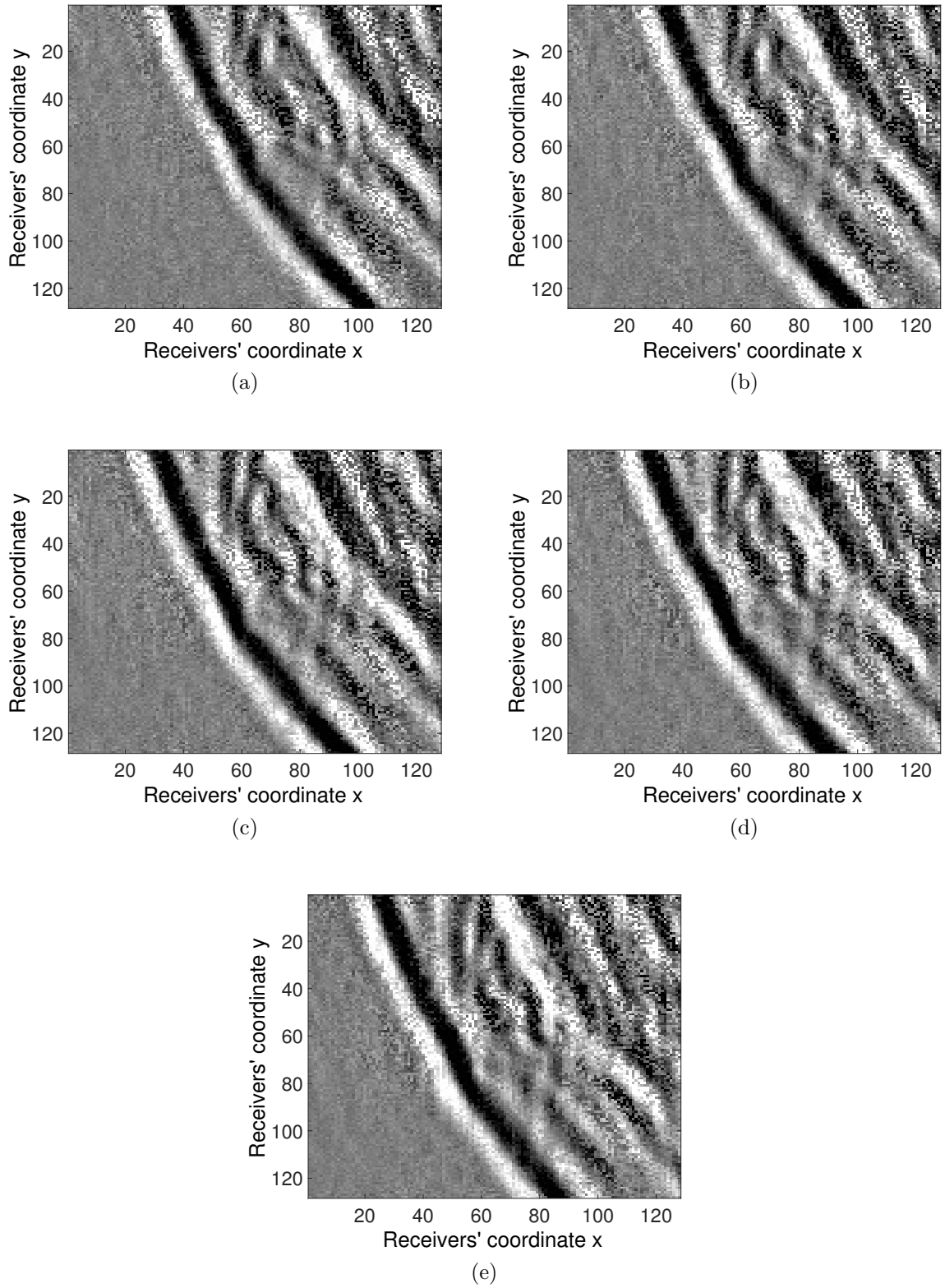
(a) (b) (c) (d) (e)

Fig. 5.26 An original 3D cube extracted from the SEAM-II data set. Five sections from this cube are displayed at different timings from $t = 68$ to $t = 72$ in (a) - (e).



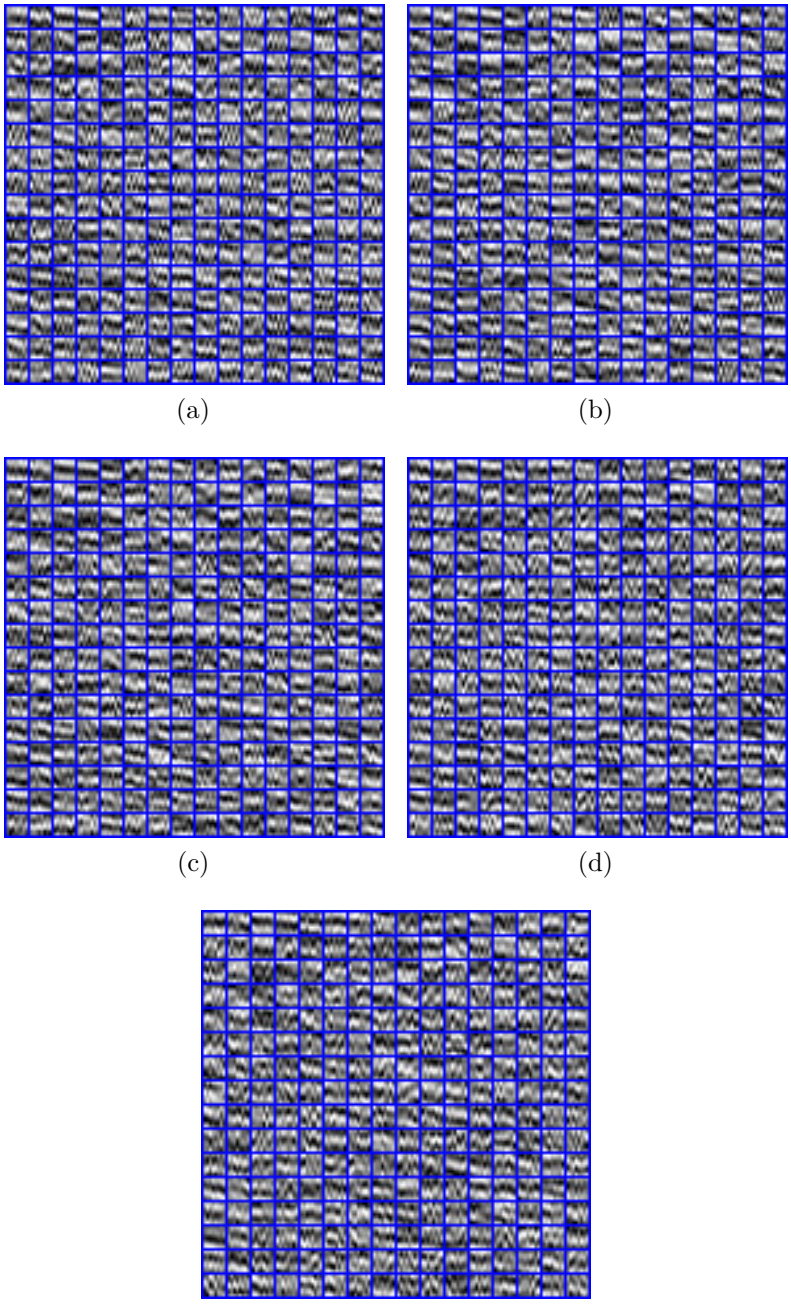
(a) (b) (c) (d) (e)

Fig. 5.27 Using only 50% of receivers from the 3D cube extracted from the SEAM-II data set. Five sections from this cube are displayed at different timings from $t = 68$ to $t = 72$ in (a) - (e).



(a) (b) (c) (d) (e)

Fig. 5.28 BPFA reconstruction from 50% of the 3D cube. Five sections are displayed from $t = 68$ to $t = 72$ in (a) - (e). Poor reconstruction due to small number of iterations. Increasing the number is impractical.

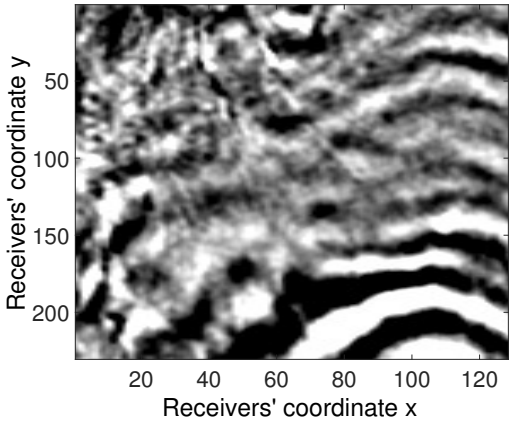


(a) (b) (c) (d) (e)

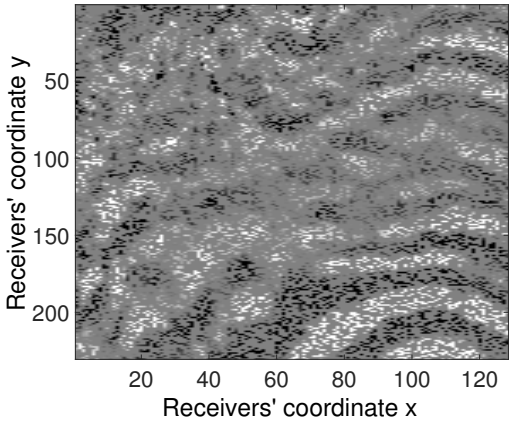
Fig. 5.29 The corresponding BPFA bases for $t = 68$ to $t = 72$ from the 3D cube in Figure 5.28. Some bases are coherent over time showing the evolution as the time increases. Others are incoherent and are similar to noise due to the lack of iterations.

5.12 Field data set

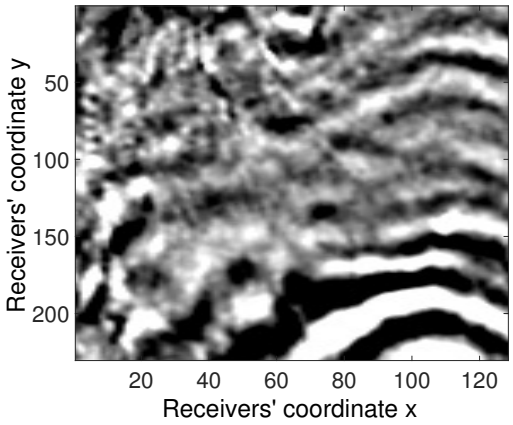
Finally, we wanted to test BPFA on a field data set. The Parihaka data set (SEG, 2018b) is a 3D seismic image described earlier in section 2.1. We first illustrate a section from a time slice of this data set in Figure 5.30(a). The same signal using 50% of receivers is shown in Figure 5.30(b) and the respective BPFA reconstruction in Figure 5.30(c). The learned bases are shown in Figure 5.30(d). Combining many sections, Figure 5.31(a) shows an entire time slice with BPFA reconstruction in Figure 5.31(c) from only 30% of receivers. The reconstruction was obtained by splitting the slice into smaller sections (128×128, 128×154 for the upper right part, 230×128 for the lower part and 230×154 for the right lower corner). From each section, individual basis functions were learned. Using PCA as in section 5.7, the feature space in Figure 5.31(e) was obtained.



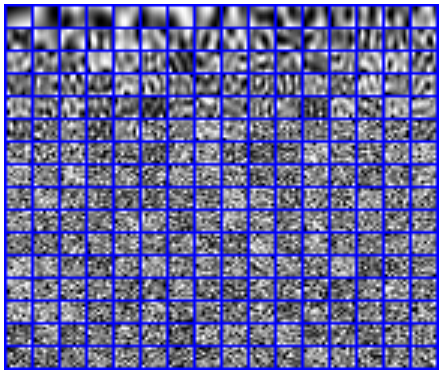
(a) Original signal



(b) Using 50% of receivers

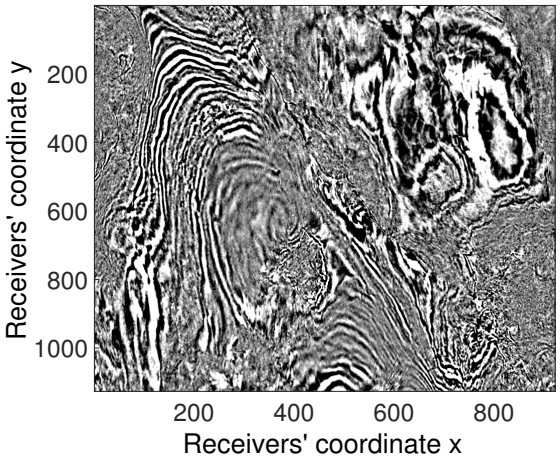


(c) BPFA's reconstruction

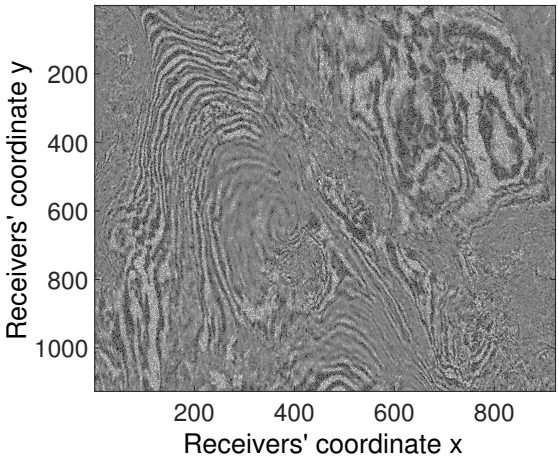


(d) BPFA's bases

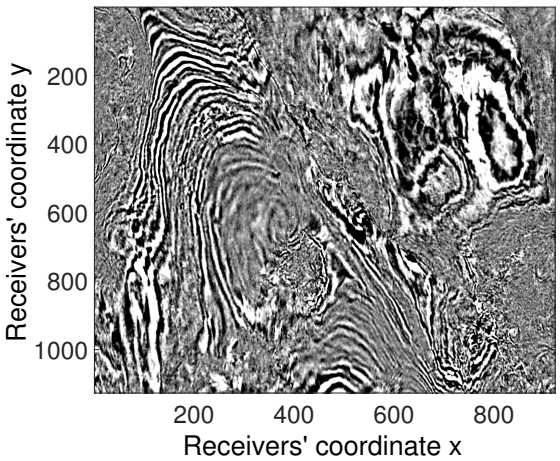
Fig. 5.30 Original (a), using 50% (b), BPFA's reconstruction (c) and learned bases (d).



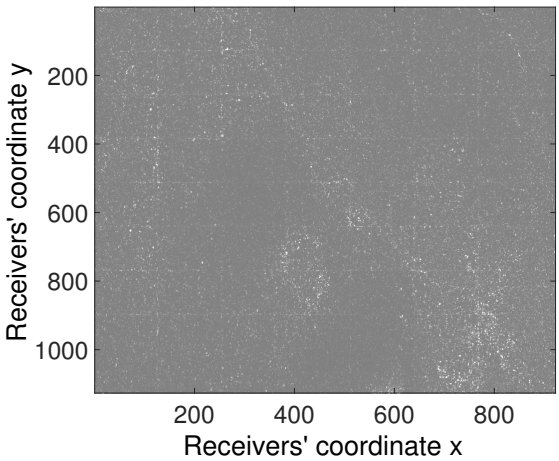
(a) Original



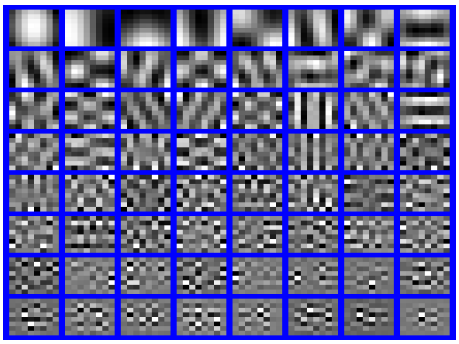
(b) Using 30% of receivers



(c) BPFA



(d) Error difference



(e) Learned basis functions over the entire time slice from 14336 basis functions

Fig. 5.31 BPFA reconstruction of a time slice at $t = 923$ from the field data set of Parihaka.

5.13 Denoising using the BPFA model

Learning a data-driven model for seismic data is not only useful for prediction of missing receivers, but it also allows further extensions into noise attenuation and denoising. Prediction and denoising are very similar since the former predicts missing receivers and denoising predicts receivers that are there but are contaminated by noise. The task of denoising is to estimate the level of noise and to choose the appropriate basis functions with the correct coefficients that correspond to the noise-free signal. Usually the dictionary is pre-fixed and chosen in order to provide a sparse representation. This is similar to the Relevance Vector Machine (RVM) model where we briefly investigated its denoising capabilities but with no success due to the lack of learning bases.

In this section, we propose to use Beta Process Factor Analysis (BPFA) to learn the dictionary of bases and denoise. We used the SEAM-II data set and added Gaussian noise with increasing levels of distortion. We extracted two hundred sections of size 128×128 from time slices of varying structures and two hundred sections of size 128×128 from seismic signals in the x-t domain. In this case, we can work on signals in the x-t domain directly since there are no missing data (i.e. big gaps) but rather noise. We compared BPFA against the K-Singular Value Decomposition (SVD) (Aharon et al., 2006) which has shown success in seismic denoising (Turquais et al., 2015; Zhu et al., 2015). The K-SVD results were produced using the package from one of the authors' website [2] and the BPFA results from the same source as mentioned in section 5.3.

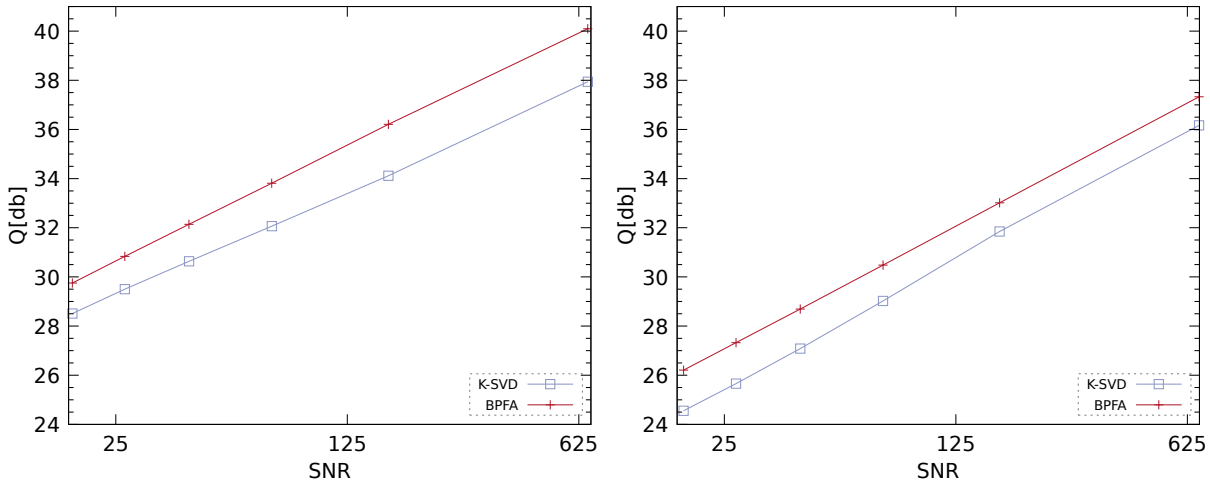
Different levels of noise in the seismic signals translate to varying Signal-to-Noise Ratio (SNR). There are numerous definitions of the SNR and it is difficult to compare between studies. However, in our case, the important value is not the SNR but rather the quality of reconstruction accuracy, Q, of each algorithm and how it compares with the other. In our experiments, we varied the noise variance to control this ratio and we define it as it was done in the denoising study (Kazemi et al., 2016) with

$$\text{SNR} = \frac{\alpha_{rms}^2}{\sigma_n^2}, \tag{5.29}$$

where α_{rms} is the root mean square amplitude of the noise-free signal and σ_n^2 is the noise variance. Our experiments were undertaken over multiple seismic signals and thus, the mean SNR was calculated over all signals. Six different values of the noise variance were used, resulting in six different mean SNR values for time slices and six for signals in the

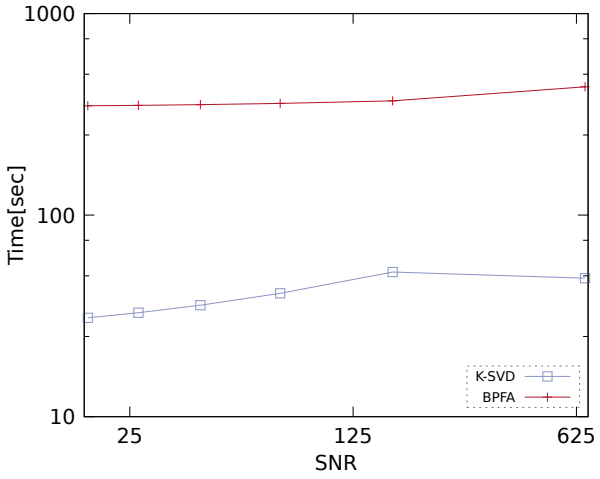
[2] Elad, M., 2006, K-Singular Value Decomposition (SVD) software, http://www.cs.technion.ac.il/~elad/software/, accessed 4 May 2016.

x-t domain. To evaluate the reconstruction, we define the quality as it was defined in equation 4.32. The mean Q for all sections is plotted against varying mean SNR values. It can be seen in Figures 5.32(a) and 5.32(b) that the BPFA attains higher levels of Q than the K-SVD for all SNR, illustrating its superiority. Since accuracy is not always enough, the computational time is shown in Figure 5.32(c) with the K-SVD being faster than the BPFA. An example of time slice denoising by both algorithms is given in Figure 5.33 and an example of denoising of signals in the x-t domain can be seen in Figure 5.34. The BPFA learns a dictionary of bases with more high frequency characteristics as opposed to the K-SVD resulting in higher accuracy from those bases.



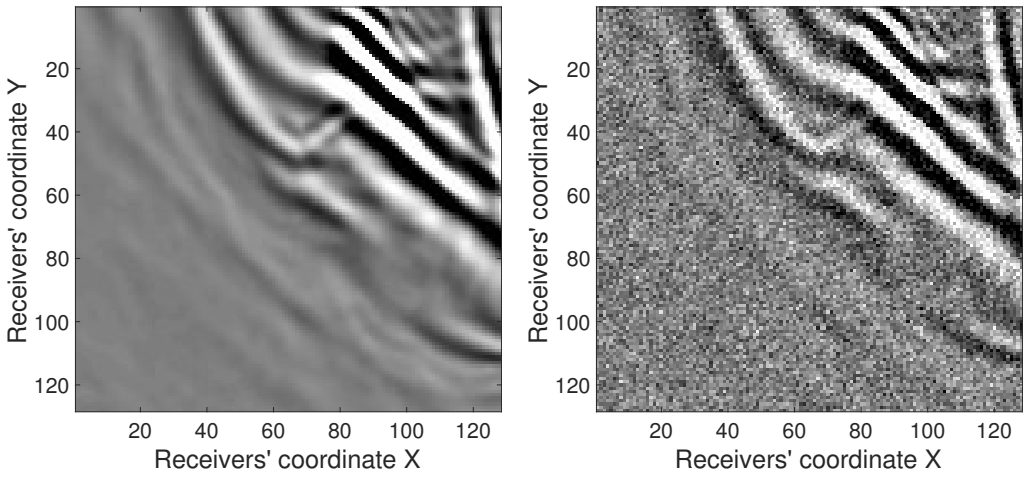
(a) Mean Q for time slices

(b) Mean Q in the x-t domain

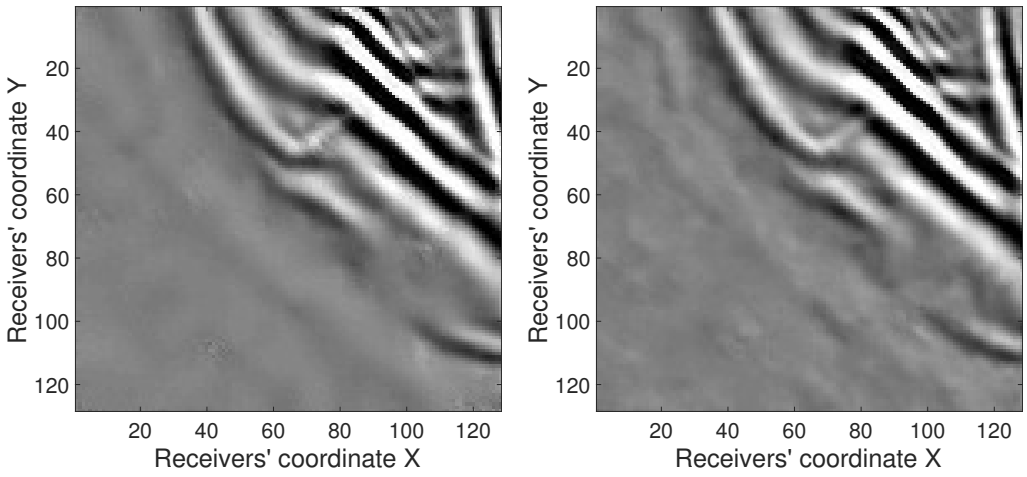


(c) Mean computational time

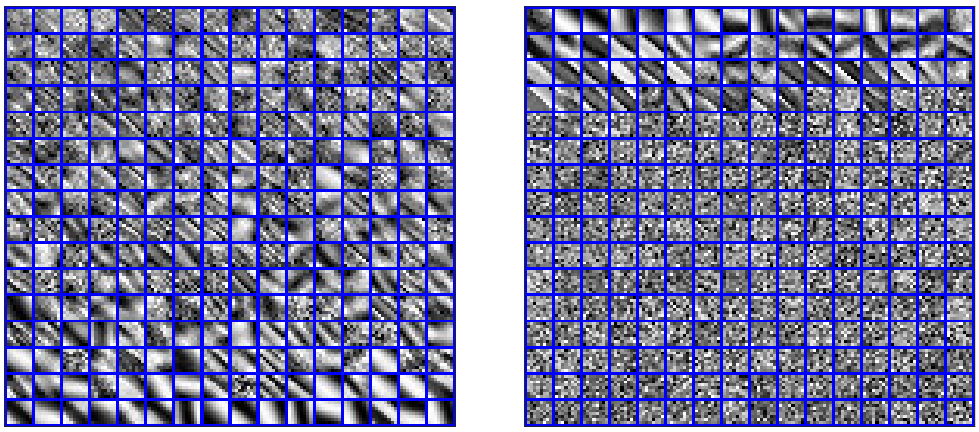
Fig. 5.32 Mean reconstruction accuracy for time slices (a), for x-t domain (b) and mean computational time (c) for time slices for two hundred sections of 128×128 per domain with varying SNR.



(a) Original (b) SNR = 20.67

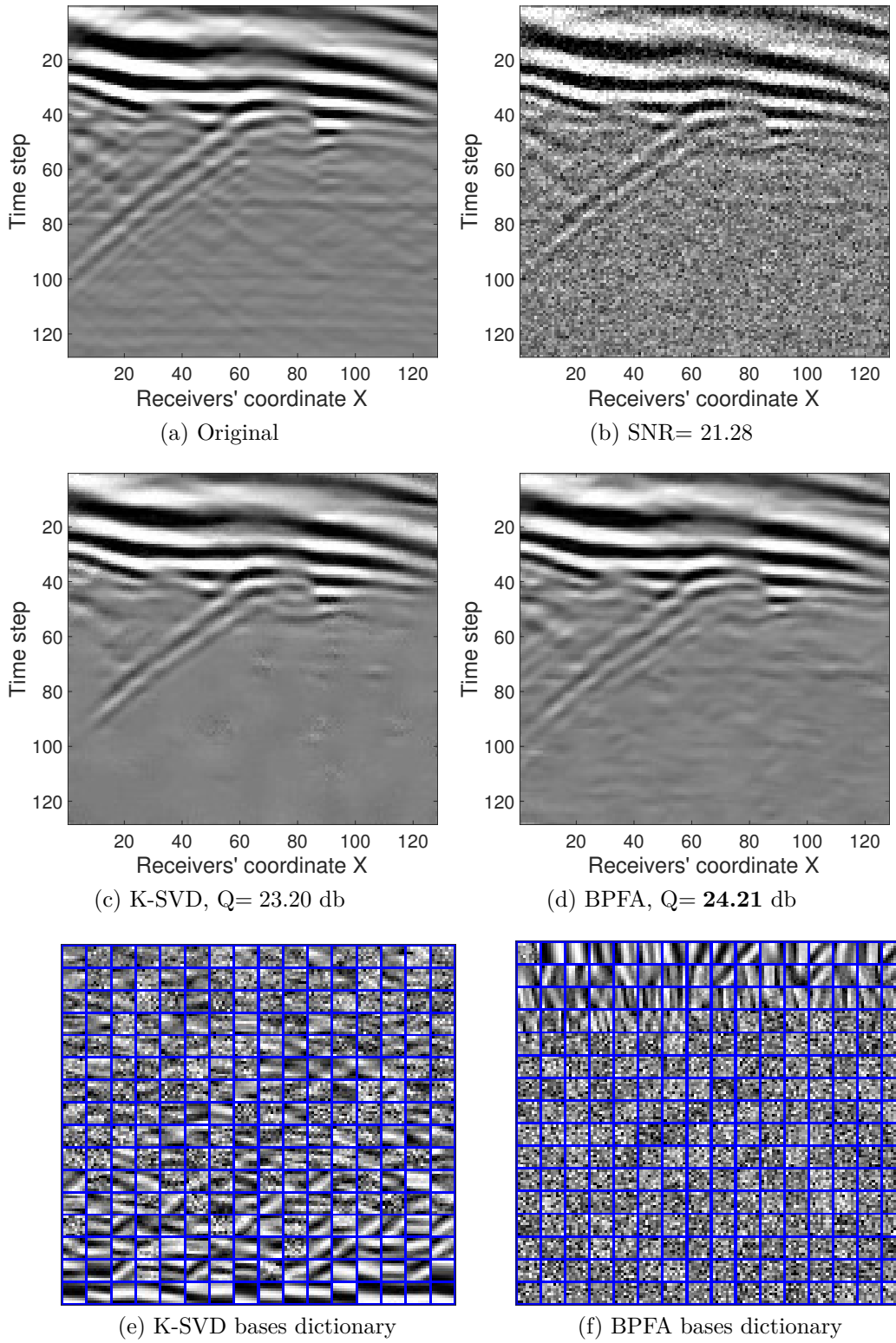


(c) K-SVD, Q= 24.25 db (d) BPFA, Q= **24.82** db



(e) K-SVD bases dictionary (f) BPFA bases dictionary

Fig. 5.33 A section of a time slice from the SEAM-II (a) is corrupted (b). BPFA (d) obtains better quality than the K-SVD (c). The learned bases dictionary of K-SVD (e) and BPFA (f) are shown. BPFA puts greater emphasis on higher frequencies. We use Q as defined in equation 4.32.



(a) Original (b) SNR= 21.28

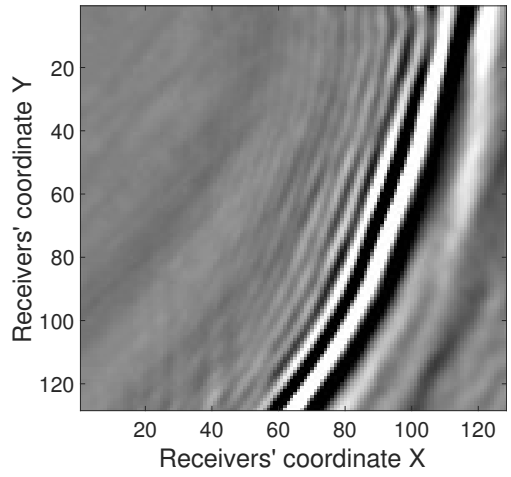
(c) K-SVD, Q= 23.20 db (d) BPFA, Q= **24.21** db

(e) K-SVD bases dictionary (f) BPFA bases dictionary

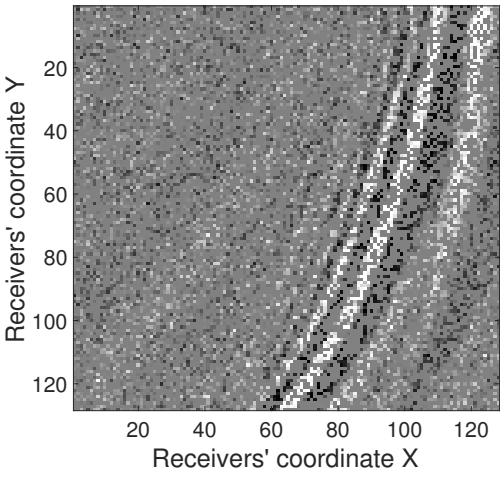
Fig. 5.34 A section of a signal in the x-t domain from the SEAM-II (a) corrupted (b). BPFA (d) obtains higher reconstruction quality than K-SVD (c). The learned bases dictionaries of K-SVD (e) and BPFA (f) are also illustrated. We use Q as defined in equation 4.32.

Simultaneous denoising and interpolation

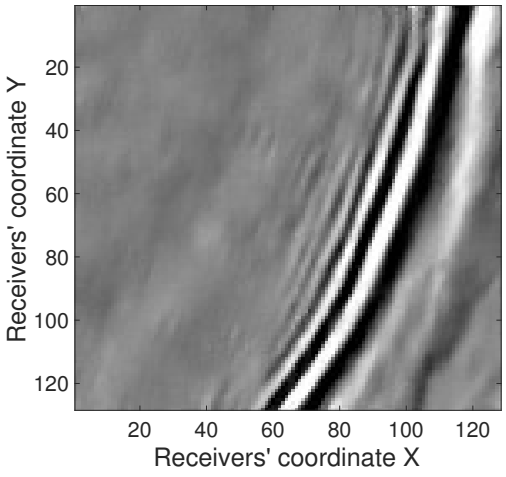
To illustrate the effectiveness of learning a data-driven model, we examine the reconstruction when 50% of receivers are missing and the rest are corrupted by noise with SNR= 20.84. Figure 5.35(a) shows the original section, Figure 5.35(b) shows the corrupted signal and Figure 5.35(c) shows the BPFA reconstruction. Figure 5.35(d) shows the dictionary of bases learned from the available data. The BPFA model is very adaptable even in very challenging scenarios for time slices. In the next chapter, we will examine its behaviour in the x-t domain and whether it is able to reconstruct without any signs of aliasing or noise in the FK domain.



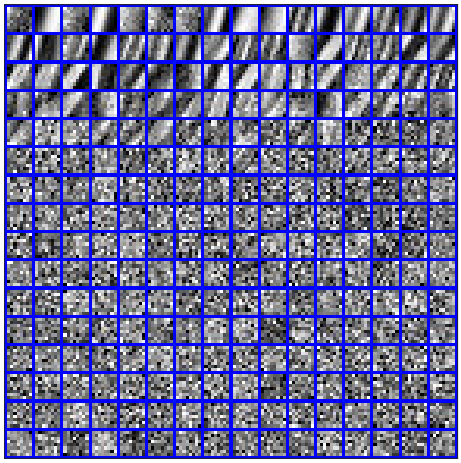
(a) Original



(b) SNR=20.84, with 50% of receivers



(c) BPFA, Q= 23.06 db



(d) Dictionary of BPFA bases

Fig. 5.35 We show (a) an original section of a time slice, (b) with added noise and 50% of receivers used, (c) the BPFA reconstruction and (d) the learned dictionary of bases. We use Q as defined in equation 4.32.

Chapter 6

x-t domain reconstruction and seismic variance analysis

One of the criteria for accurate signal reconstruction requires the visualisation of the Frequency-Wavenumber (FK) domain that we discussed in section 2.2. Using this domain, we are able to interpret whether the signal is reconstructed without any signs of aliasing (i.e. if the sampling was sufficient to capture all the details in the signal for further processing) or if there is any incoherent noise. In order to obtain the FK domain, it is necessary to use the x-t domain instead of the time slice domain as discussed in chapter 2. In this chapter, we will use the reconstructions from time slices and sort them into the x-t domain. We will evaluate these reconstructions for all algorithms and configurations and inspect their FK domain for any aliasing or noise. In addition, we will provide analysis of the results using the variance of the available data of signals and split the signals into regions for which different algorithms and configurations perform better.

6.1 Reconstructions with time slice processing and the x-t domain

We have shown in Figure 4.5 that the RVM obtains poor reconstruction when working directly with large gaps. We will also illustrate with an experiment that the BPFA does not perform well when operating directly in the x-t domain due to large consecutive gaps in the training data. Figure 6.1 shows the x-t domain with only 30% of the receivers being used. If we try to reconstruct this directly, there are large gaps from consecutive missing receivers as seen in Figure 6.2. Figure 6.2(a) shows part of the original signal of Figure 6.1.

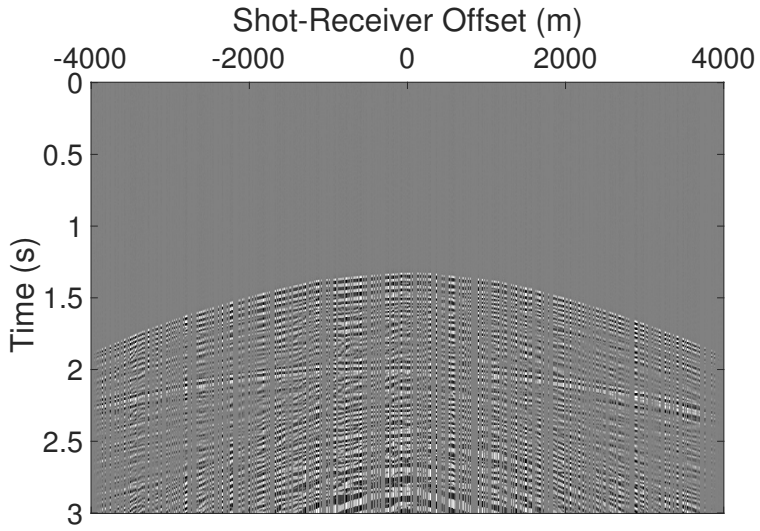
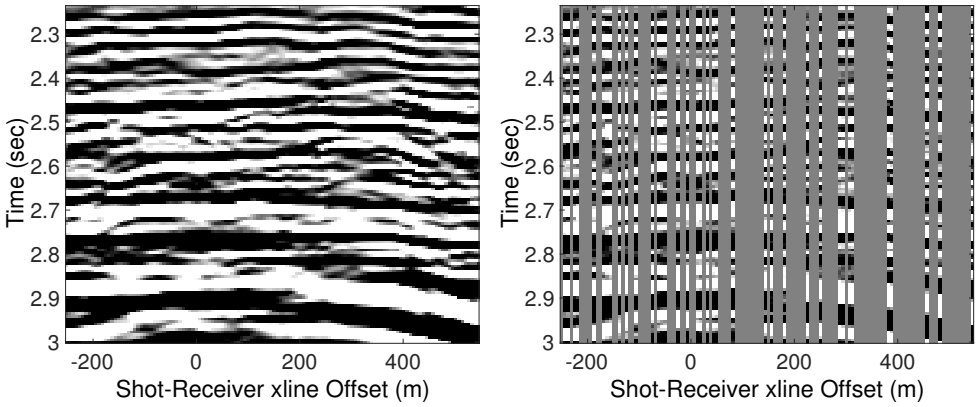
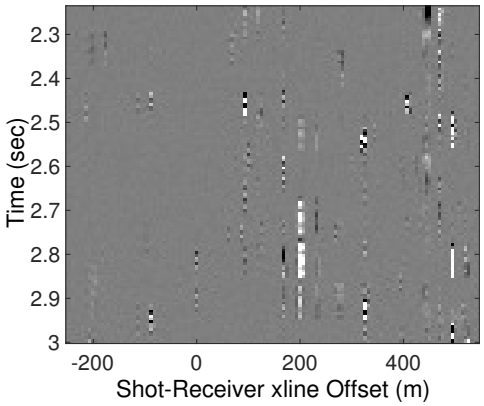


Fig. 6.1 Using 30% of the receivers in the x-t domain.



(a) Close up region of original

(b) Close up region with missing traces



(c) BPFA failed reconstruction of (b)

Fig. 6.2 Parts of the signal of Figure 6.1. It is not possible to include the reconstruction in one plot since the range of values varies by many orders of magnitude.

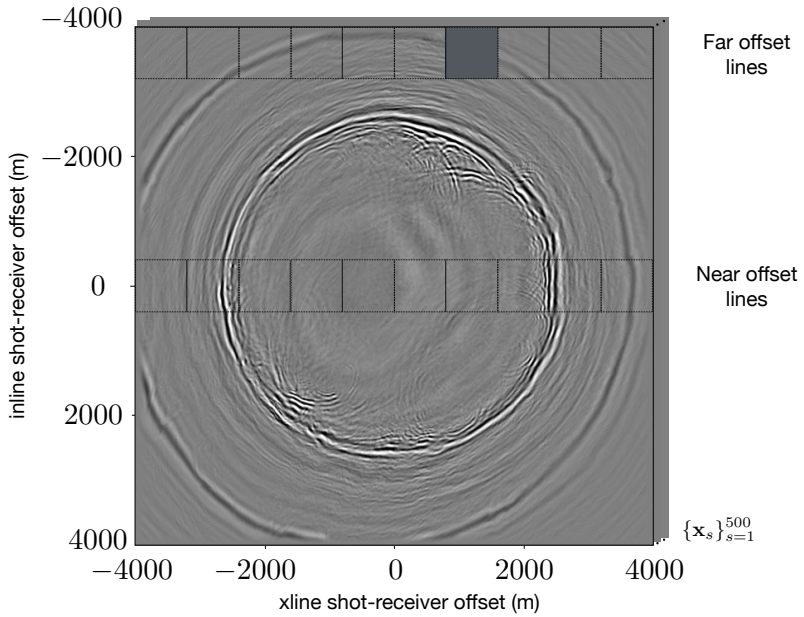
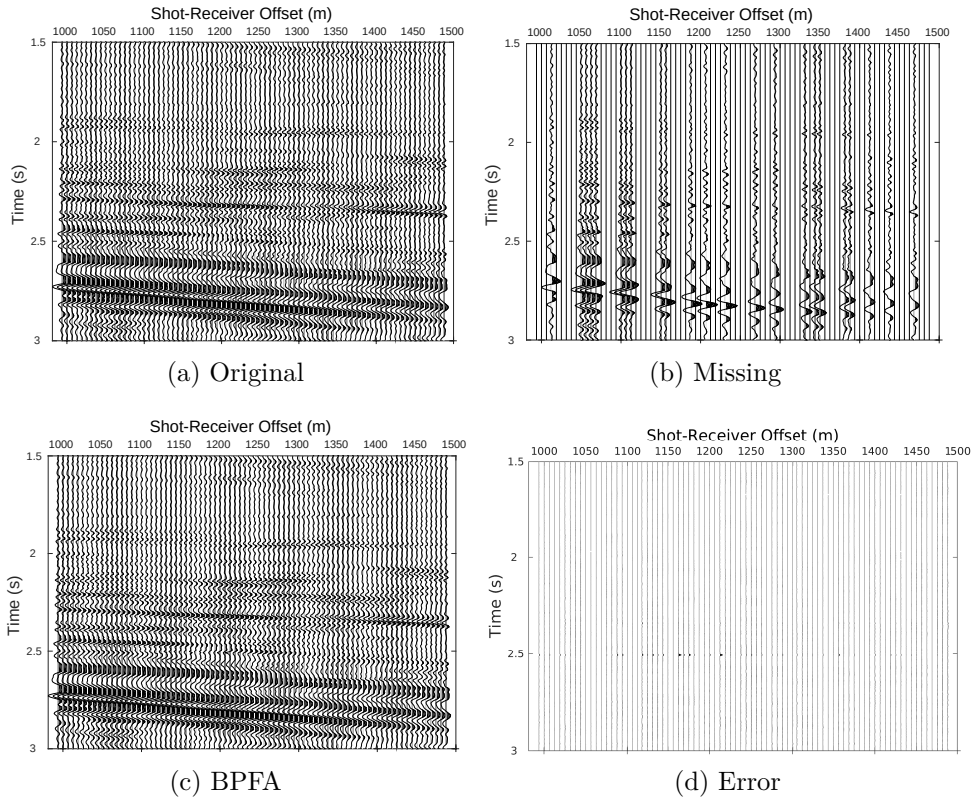


Fig. 6.3 Sections from far and closer to the source were extracted over all time samples.

Figure 6.2(b) shows part of the signal of Figure 6.1 and we can see that there are large gaps in the data and the reconstruction in Figure 6.2(c) is poor. This is due to consecutive receivers missing and the BPFA inference does not have sufficient representative examples to capture the underlying structure of the signal.

With this in mind, we decided to operate in the time slice domain, remove receivers corresponding to data points randomly, reconstruct each section of a time slice and then sort the data in the x-t domain. We used the same 3D synthetic data set generated numerically using the SEAM-II model as input. The data set is composed of one source with a 1281×1281 receiver grid and spatial sampling at 6.25 metres. The sampling rate in time is 6ms with a total of 500 time samples. Therefore, 500 time slices were extracted and from each, two sets of 10 sections of 128×128 (the last section is 128×129). The sections were extracted close to and far from the source (as shown in Figure 2.6 and Figure 6.3) in order to test the reconstruction with different signal structures. To perform irregular under-sampling as discussed in section 2.2.2, we created three masks of size 128×1281 with different percentages of receivers kept randomly (30%, 50% and 70%). These masks were fixed throughout all time slices to match with entire receivers missing.

Before discussing the results of the experiments, we provide a BPFA reconstruction in the x-t domain after re-sorting sections of time slices and zooming in to individual receivers. Figure 6.4(a) shows the original, missing receivers are set to zero in Figure



(a) Original (b) Missing

(c) BPFA (d) Error

Fig. 6.4 Reconstruction of the 127th shot line from part of the dark gray section in Figure 6.3 where 23 receivers are used from 80 original receivers.

6.4(b), the BPFA reconstruction after re-sorting in the x-t domain in Figure 6.4(c) and the reconstruction error is shown in Figure 6.4(d). Figure 6.3 includes a dark gray section for the location of Figures 6.4 (127th receiver line of section closest to source).

6.2 Comparisons for far from source receiver lines

Experiments in the x-t domain, regarding the reconstruction accuracy are included for the RVM with DCT bases, POCS and SPGL1 with DCT bases on both 8×8 and 128×128 patches. Results are also included for the BPFA on 8×8 patches and for the RVM and SPGL1 using the learned dictionary of bases from each section on 8×8 patches.

We start the comparisons with the far from source receiver lines. As we have seen in Figure 2.6, 10 sections far from the source for all time samples are extracted resulting in 5000 sections. We used 30%, 50% and 70% of the receivers and reconstructed with all algorithms. After all reconstructions are finished, we re-sort the signals in the x-t domain resulting in 128 receiver lines that are far from the source. In all results, we magnify the

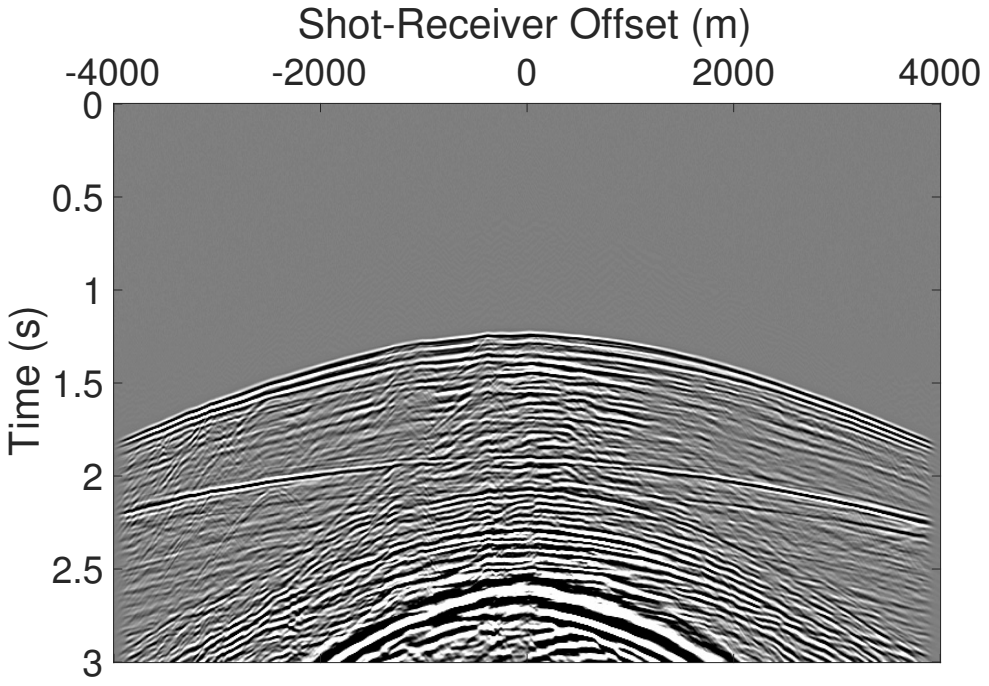
first 40 Hz of the FK domain (where most of the signal lives) in order to visualise the frequency details better.

The x-t domain of a line of receivers far from the source is in Figure 6.5(a) with its FK domain in Figure 6.5(b). Figure 6.6(a) shows only 30% of receivers. The FK domain of the signal with zeros has a lot of incoherent noise as Figure 6.6(b) shows. Using the RVM with DCT on 128×128 patches, we obtain the x-t domain in Figure 6.7(a). Its FK domain is in Figure 6.7(b) with no noise or any aliasing. The x-t domain reconstructions of POCS and SPGL1 with 128×128 patches are in Figures 6.8 and 6.9 respectively with no aliasing or noise. Nevertheless, the reconstruction quality, Q, of the RVM is higher compared to the others. Illustration of results with 8×8 are given next.

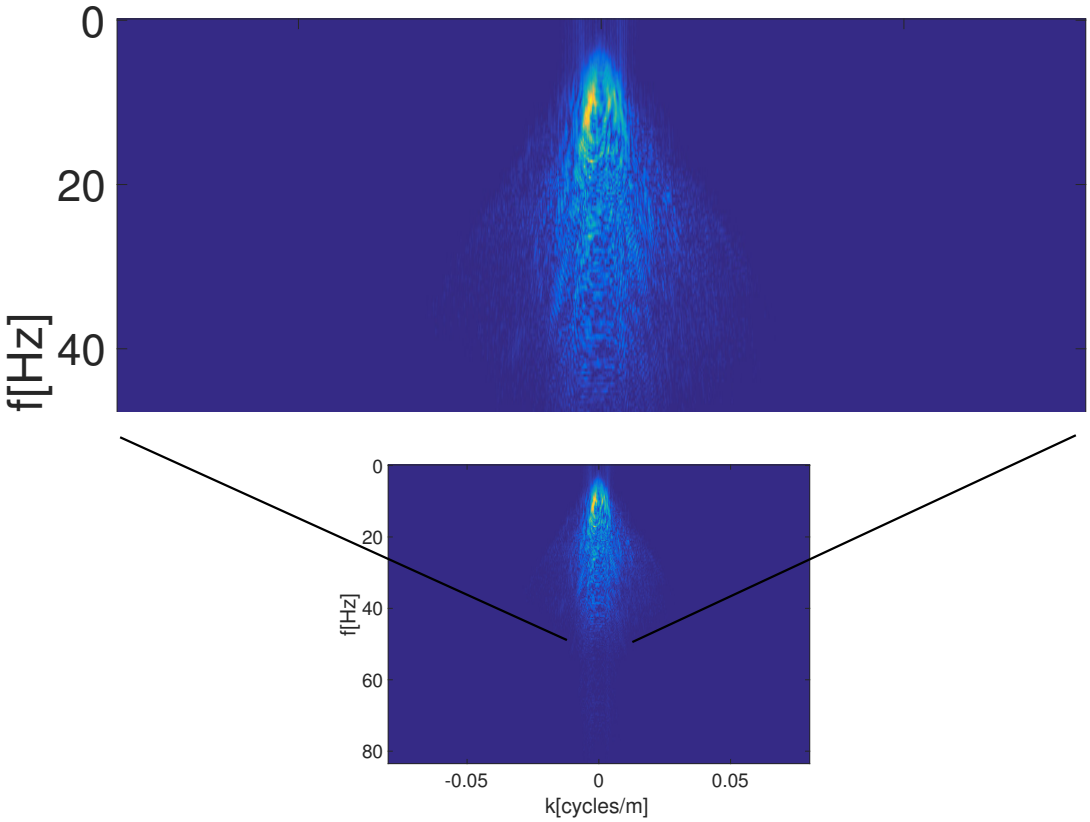
The BPFA reconstruction on 8×8 patches is given in Figure 6.10(a) with high Q. There is no signs of aliasing or noise in its FK domain in Figure 6.10(b). The POCS reconstruction on 8×8 patches is in Figure 6.11(a). This shows a reconstruction with much lower accuracy and distorted receivers. Its FK domain is in Figure 6.11(b) with signs of incoherent noise showing that the reconstruction accuracy obtained by POCS on 8×8 patches is not sufficient. The SPGL1 using the DCT on 8×8 patches is in Figure 6.12(a) which also does not perform well with noise in the FK domain in Figure 6.12(b).

By changing the SPGL1's bases to the learned dictionary of bases by BPFA per section, we improve the reconstruction as it can be seen in Figure 6.13(a). Its FK domain in Figure 6.13(b) also improves with no aliasing or noise. The same behaviour is observed with the RVM. When we use the RVM and the DCT bases, we obtain a reconstruction that is worse in Figure 6.14(a) compared to the reconstruction using the learned bases from the BPFA and the RVM in Figure 6.15(a). The FK domain of the latter in Figure 6.15(b) is also better with less noise as opposed to when using the DCT in Figure 6.14(b).

Table 6.1 summarises these results where the mean reconstruction accuracy is shown for three percentages. We can see that the RVM using DCT on 128×128 patch size performs better than all other algorithms. From the 8×8 configurations, the BPFA is the best algorithm out of the algorithms with fixed bases. It is better even from the algorithms that operate on 128×128 patches (namely the POCS and SPGL1-DCT) showing the great reconstruction accuracy possible by learning bases. Using the learned bases from the BPFA with the RVM and SPGL1, we can see improvements in accuracy. The RVM-Learned improves for all percentages when compared with the RVM using DCT bases. It even obtains better results than the BPFA for one percentage. We can also see similar improvements for the SPGL1 when using learned bases as opposed to fixed DCT.

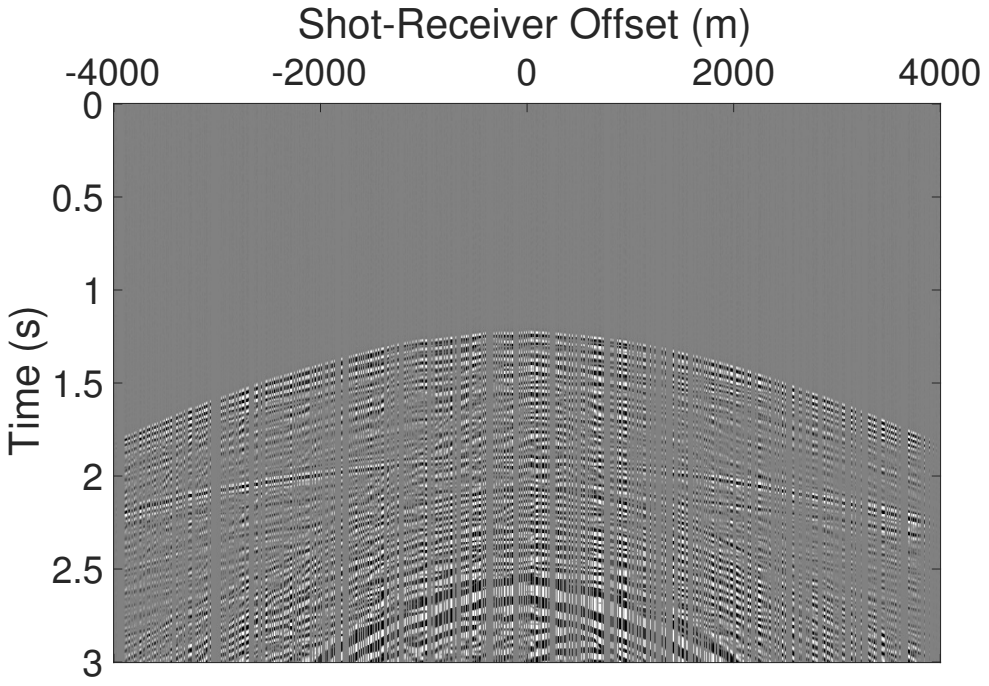


(a) Original

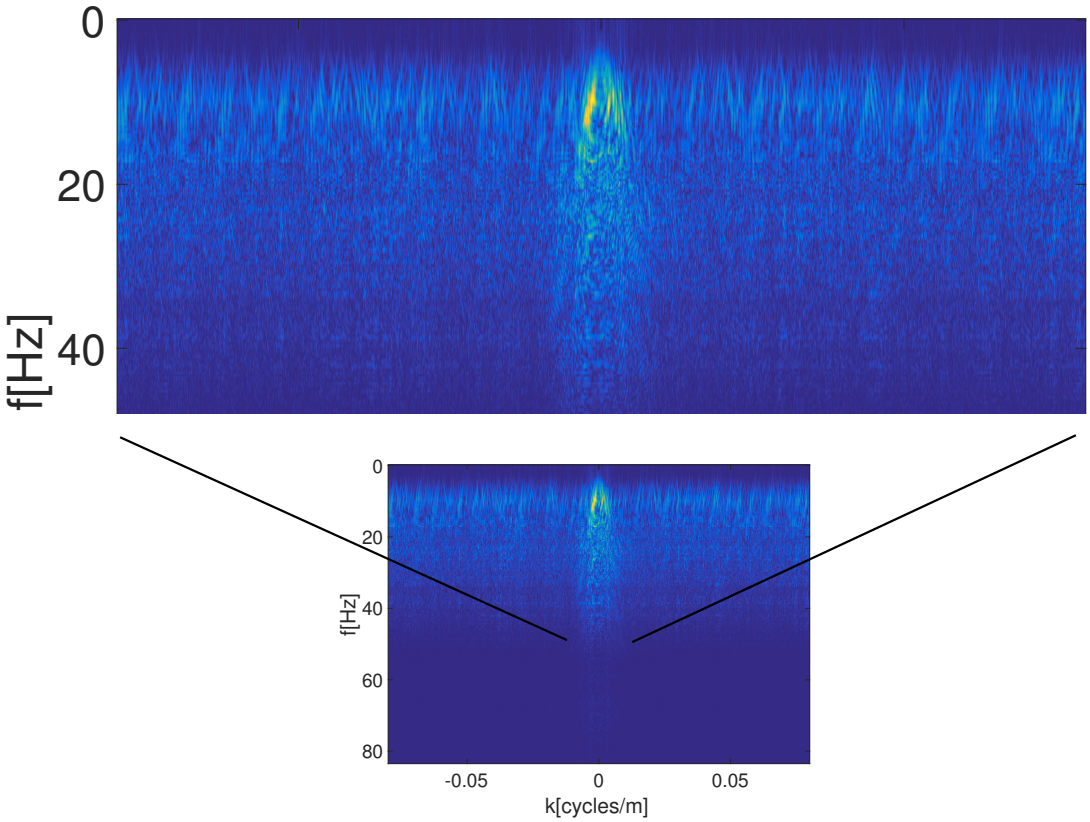


(b) FK domain of (a)

Fig. 6.5 An original receiver line (x-t domain) far from the source with its respective FK domain.

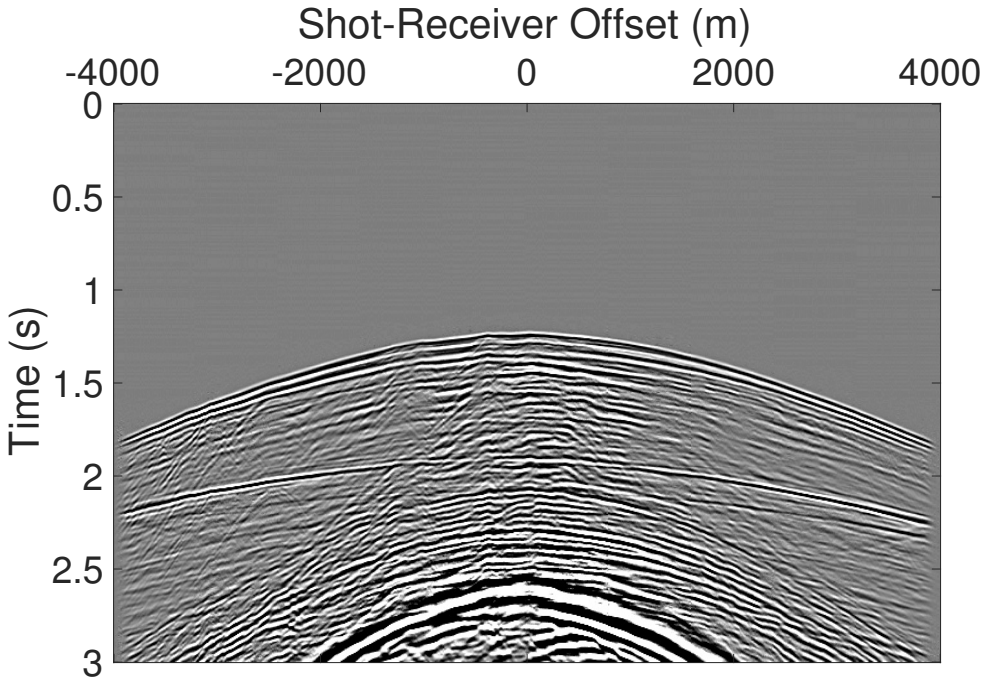


(a) Using 30% of receivers

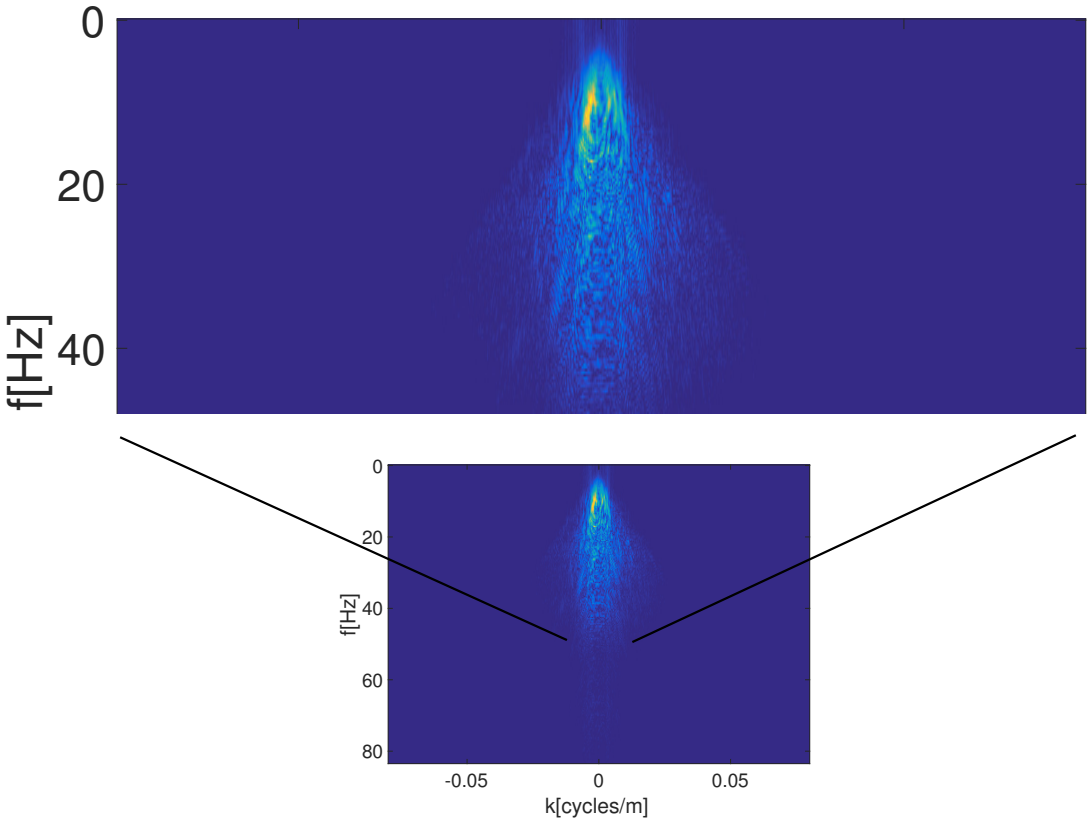


(b) FK domain of (a)

Fig. 6.6 Using 30% of receivers from a signal far from the source with its respective FK domain.

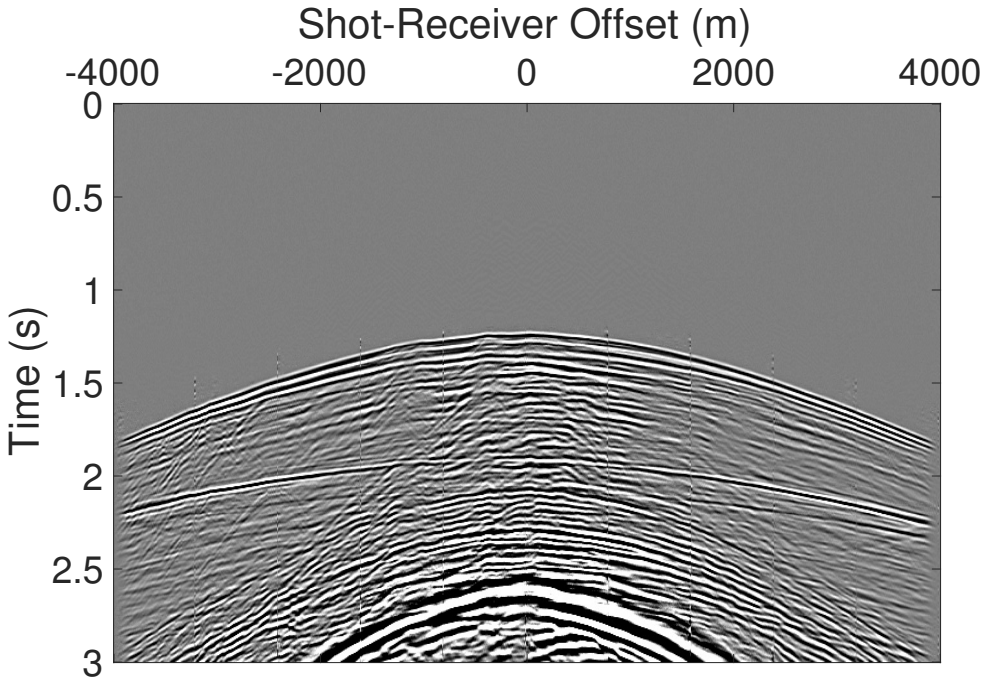


(a) RVM-DCT, 128 × 128 reconstruction, Q = 28.562 db

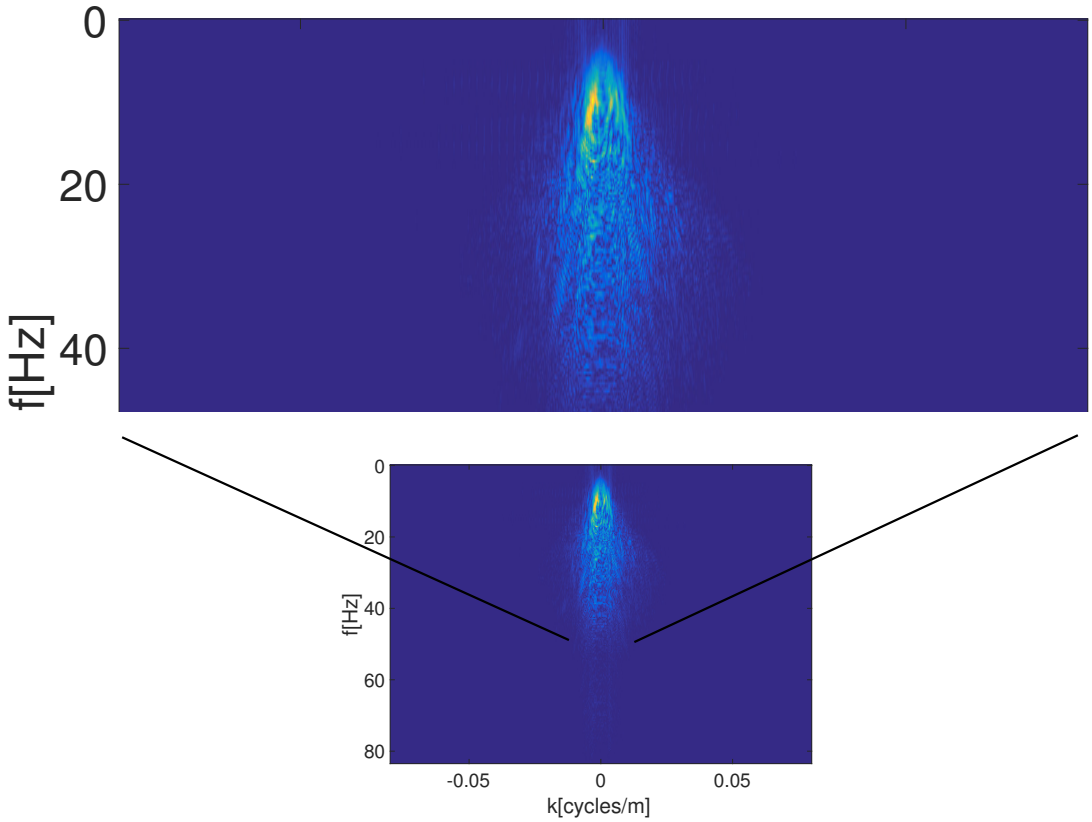


(b) FK domain of (a)

Fig. 6.7 Reconstruction using the RVM with DCT on 128 × 128 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

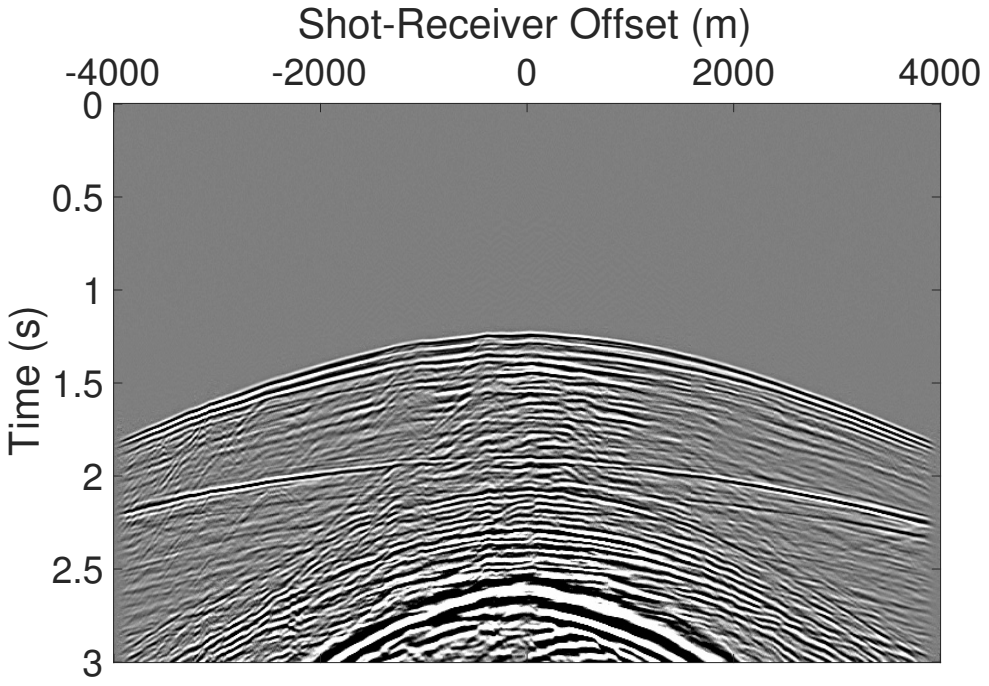


(a) POCS, 128 × 128 reconstruction, Q = 16.826 db

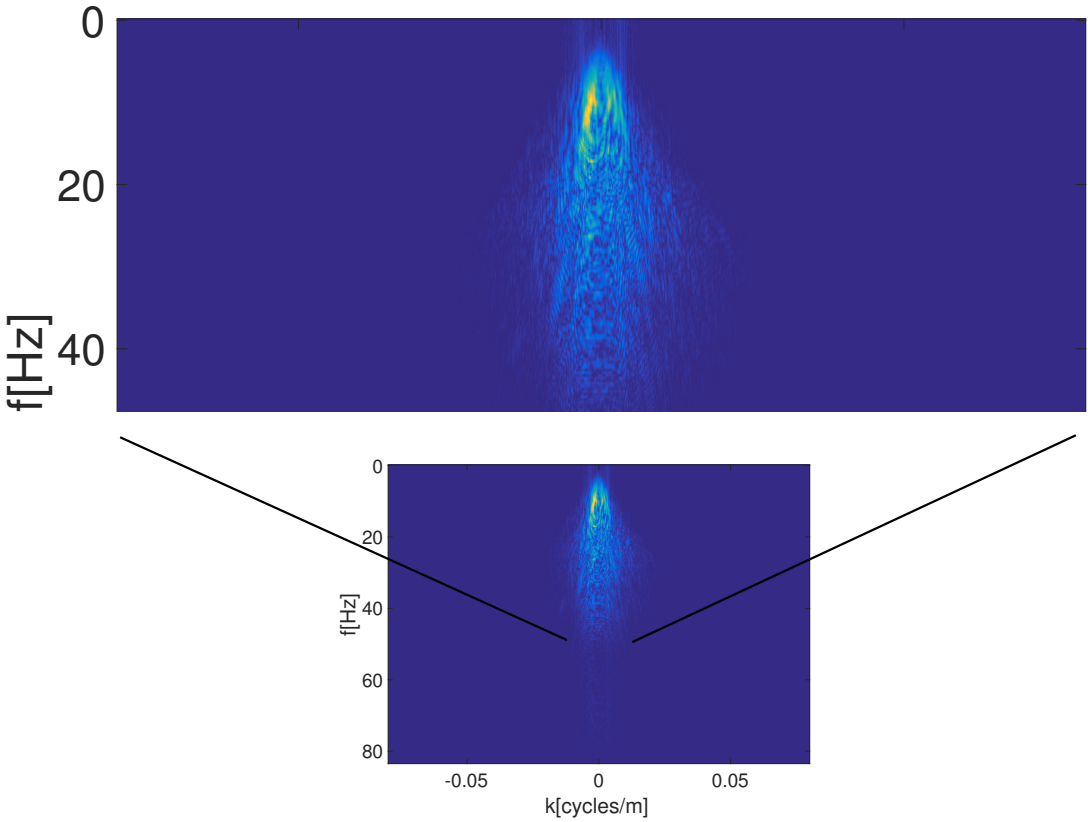


(b) FK domain of (a)

Fig. 6.8 Reconstruction using POCS on 128 × 128 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

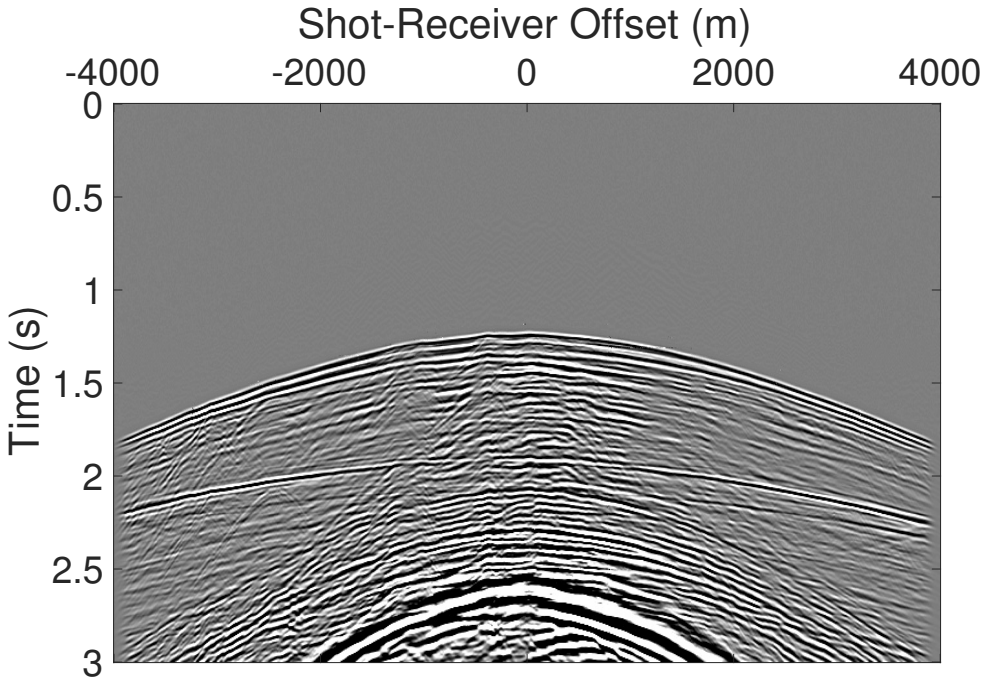


(a) SPGL1-DCT, 128 × 128 reconstruction, Q = 26.519 db

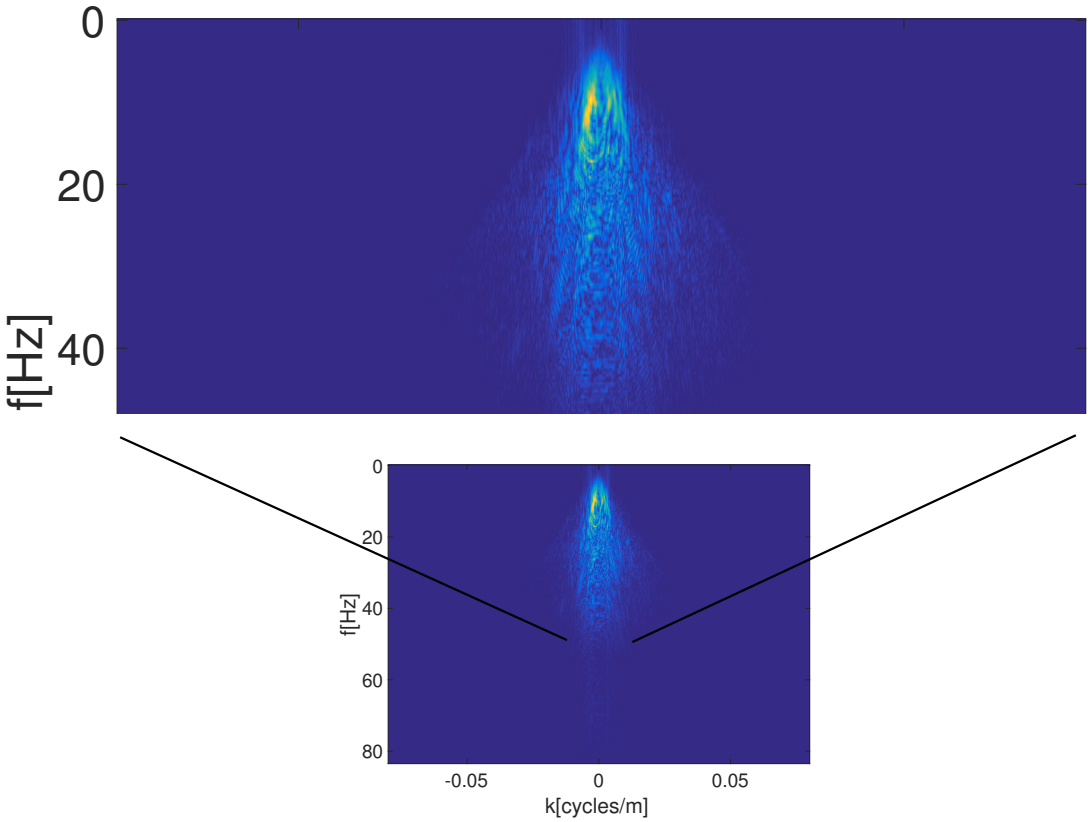


(b) FK domain of (a)

Fig. 6.9 Reconstruction using the SPGL1 with DCT on 128 × 128 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

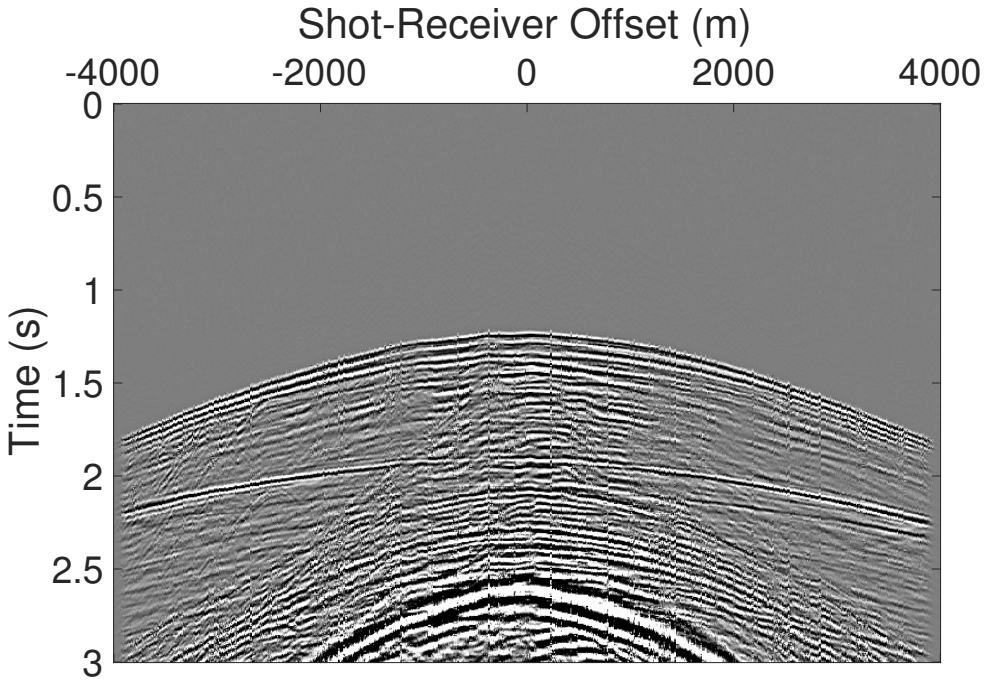


(a) BPFA, 8×8 reconstruction, Q = 23.481 db

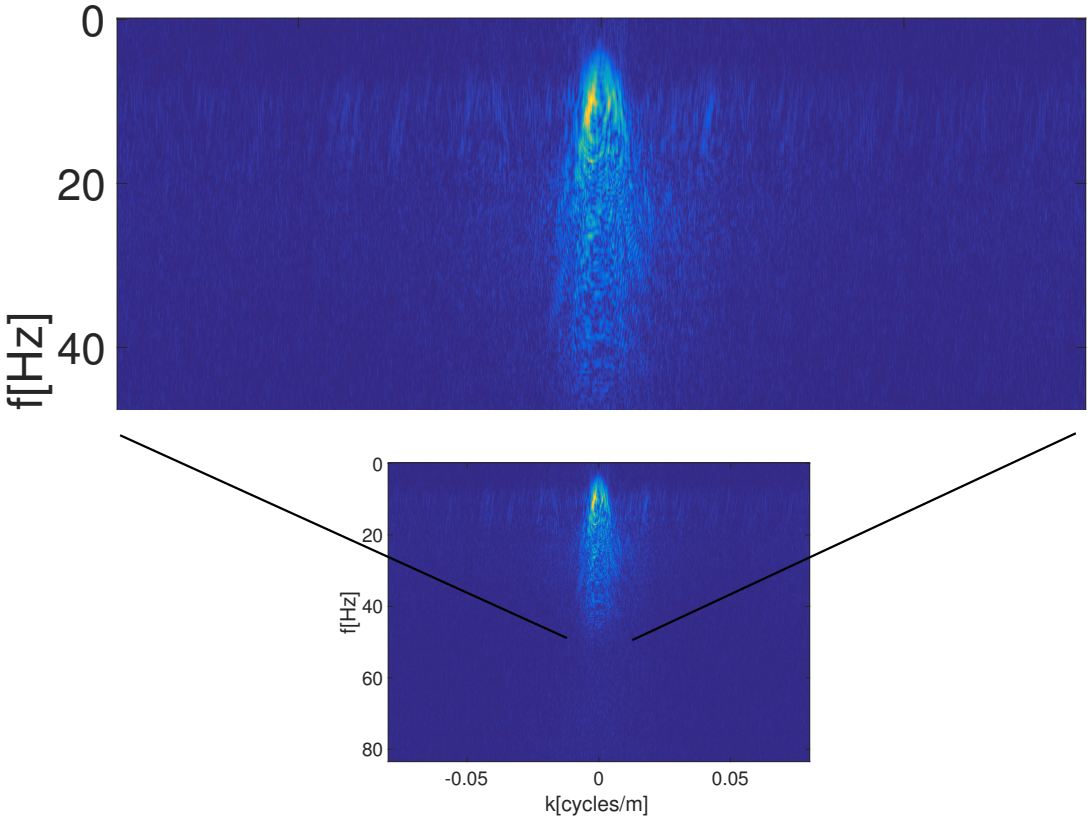


(b) FK domain of (a)

Fig. 6.10 Reconstruction using the BPFA on 8×8 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

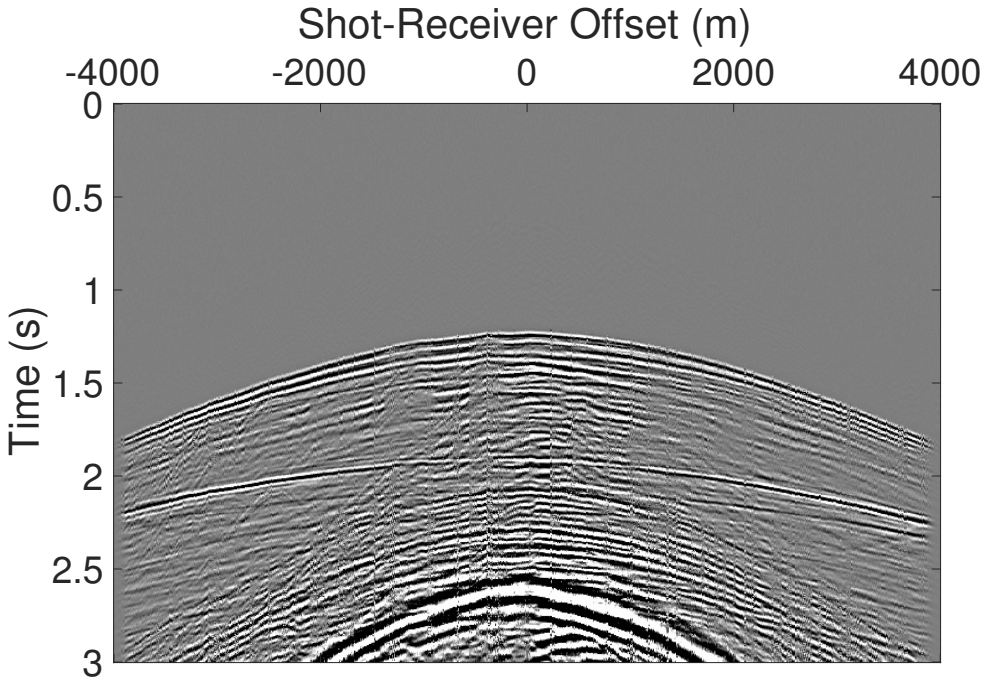


(a) POCS, 8×8 reconstruction, Q = 9.715 db

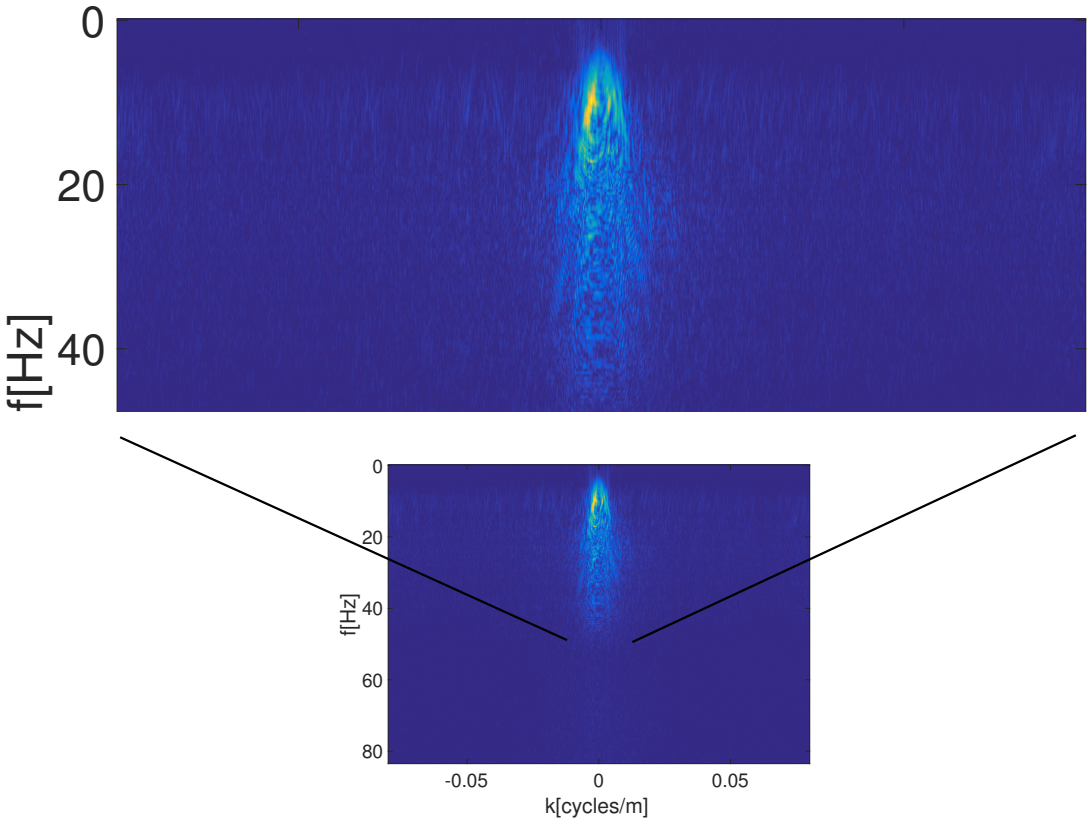


(b) FK domain of (a)

Fig. 6.11 Reconstruction using POCS on 8×8 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

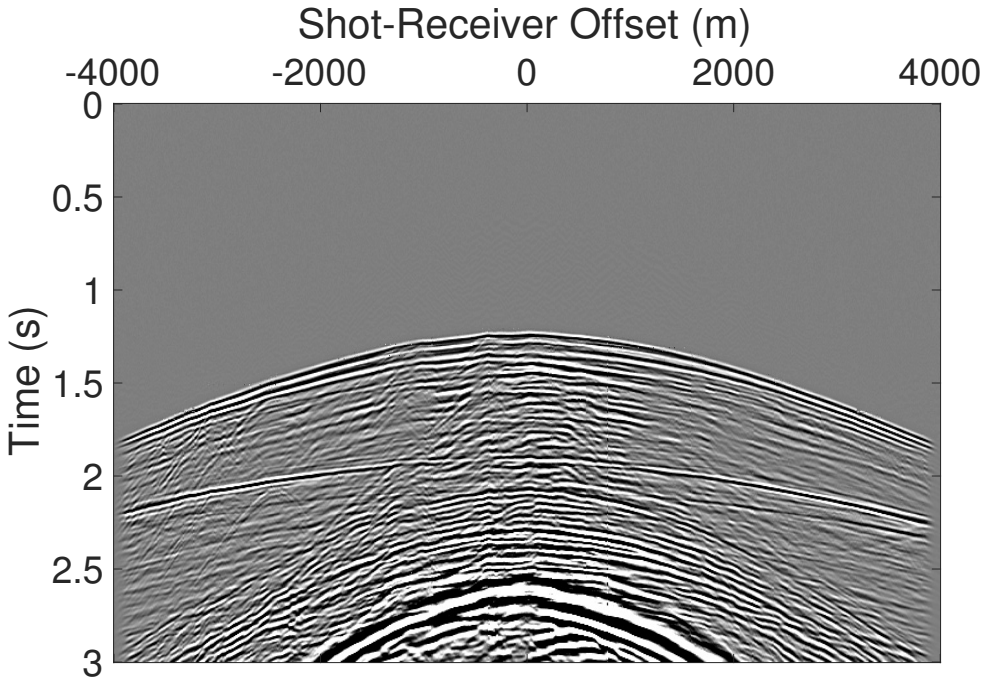


(a) SPGL1-DCT, 8×8 reconstruction, Q = 9.696 db

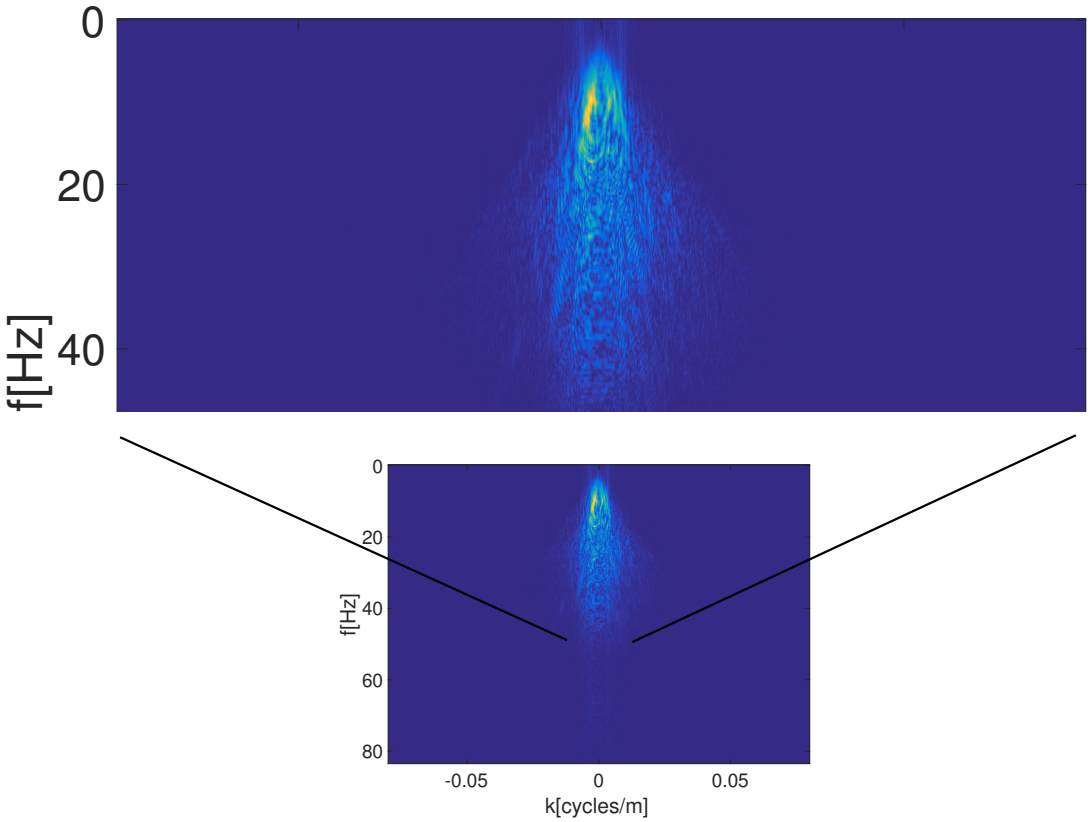


(b) FK domain of (a)

Fig. 6.12 Reconstruction using SPGL1 with DCT on 8×8 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

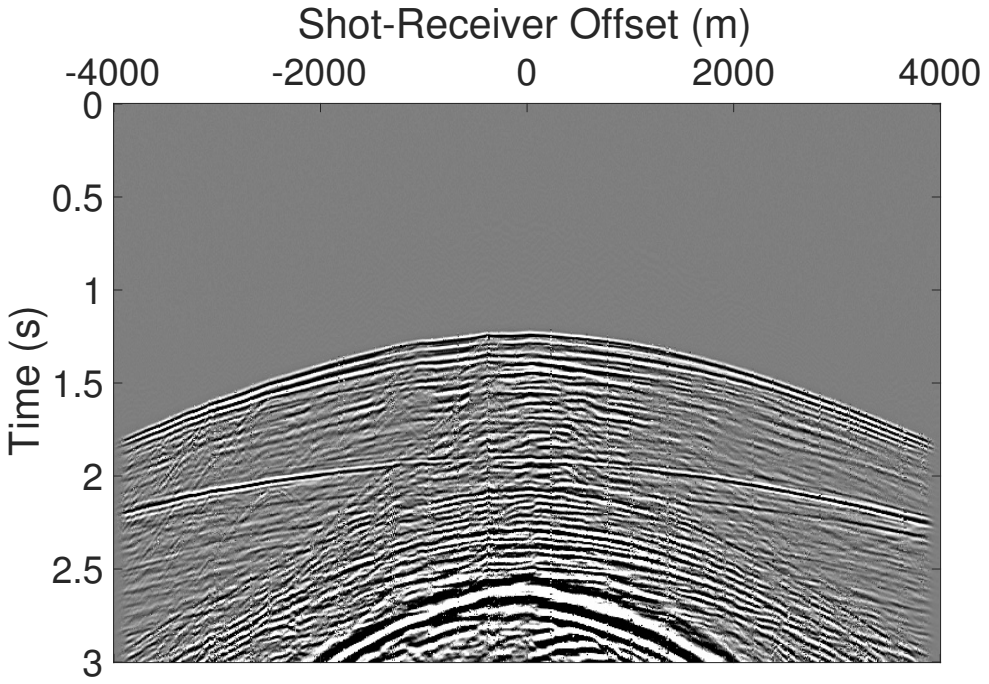


(a) SPGL1-Learned, 8×8 reconstruction, Q = 11.305 db

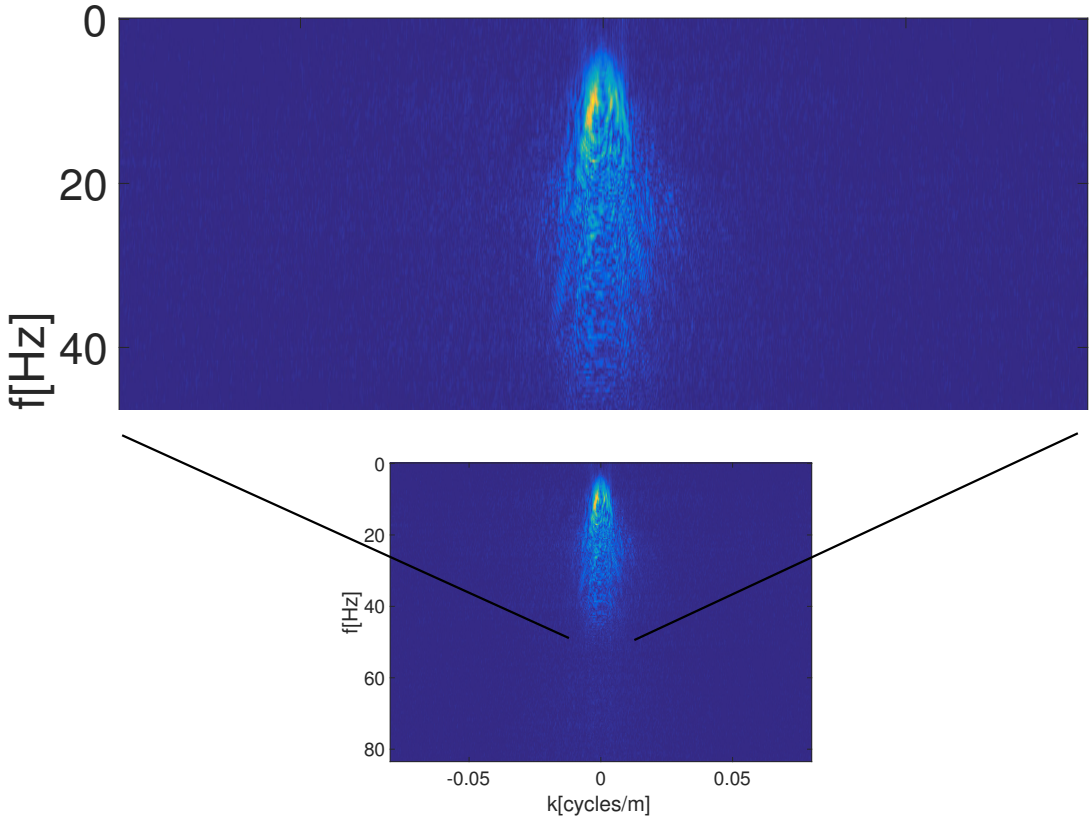


(b) FK domain of (a)

Fig. 6.13 Reconstruction using SPGL1 and learned bases on 8×8 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

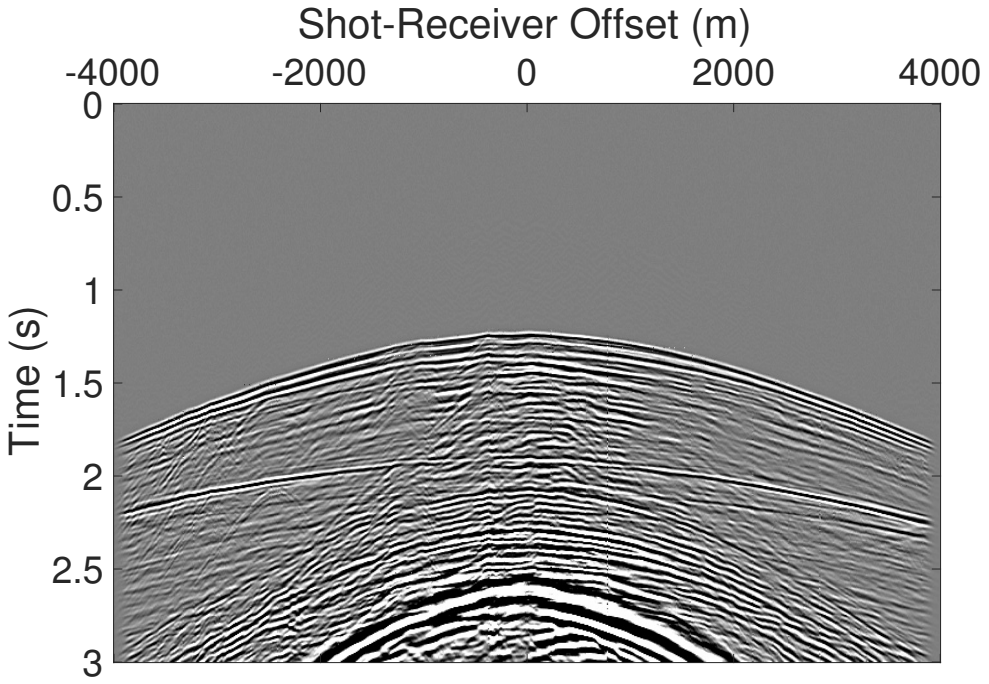


(a) RVM-DCT, 8×8 reconstruction, Q = 12.324 db

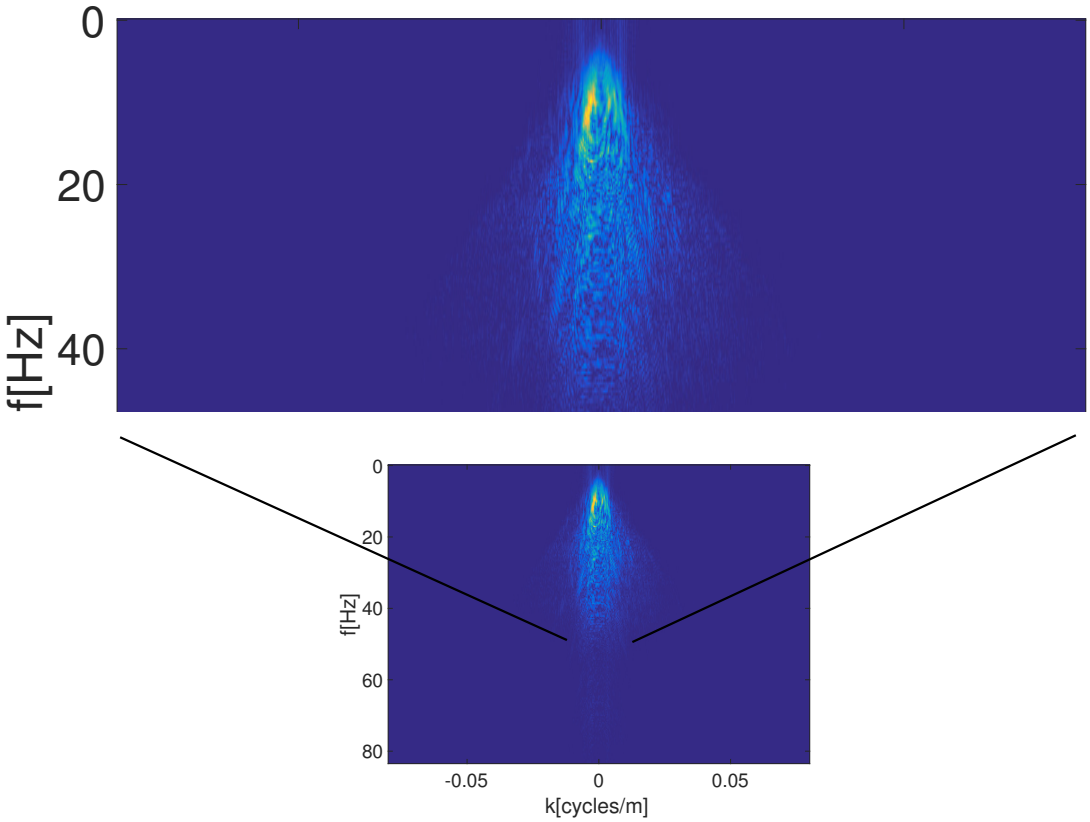


(b) FK domain of (a)

Fig. 6.14 Reconstruction using RVM with DCT on 8×8 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.



(a) RVM-Learned, 8×8 reconstruction, Q = 14.591 db



(b) FK domain of (a)

Fig. 6.15 Reconstruction using the RVM and learned bases on 8×8 patches from 30% of receivers and its respective FK domain. We use the reconstruction accuracy, Q, as defined in equation 4.32.

Far from source - Reconstruction accuracy in Q [db]			
Percentage used	30%	50%	70%
RVM-DCT 8×8	14.096	24.522	35.092
RVM-Learned 8×8	16.850	34.354	41.279
RVM-DCT 128×128	**23.569**	**35.460**	**43.679**
POCS 8×8	9.917	15.981	21.106
POCS 128×128	15.841	20.882	27.740
SPGL1-DCT 8×8	9.983	20.143	30.492
SPGL1-Learned 8×8	20.084	32.705	37.095
SPGL1-DCT 128×128	22.137	31.402	40.975
BPFA 8×8	11.545	34.257	42.436

Table 6.1 Mean Q for x-t domain for far from source.

6.3 Comparisons for close to source receiver lines

We will now discuss the reconstruction results for receiver lines that are closer to the source. These signals have different structure with steeper dips. We provide an example of a line of receivers close to the source in Figure 6.17(a) along with its respective FK domain in Figure 6.17(b). Figure 6.18(a) shows the same signal but using 50% of the receivers with its respective FK domain in Figure 6.18(b). It can be seen that there is incoherent noise in its FK spectrum and this should be removed during reconstruction.

Figure 6.19(a) shows the reconstruction obtained by the RVM with DCT on 128×128 and using the DCT. We can see that at the first time samples, the reconstruction is poor. This happens at the centre of the signal, closest to the source. This is due to the steepness of the signal and not the lack of data (there is signal at the grey areas, albeit very small). Essentially, the basis functions used do not contain characteristics with high frequencies that could capture these changes. Nevertheless, the FK domain in Figure 6.19(b) does not exhibit any aliasing and there is minimal incoherent noise.

Figure 6.20(a) shows the reconstruction obtained by POCS on 128×128 patches. It can be seen that again the reconstruction at the top and centre of the signal is poor. Overall, the reconstruction is worse than the RVM as shown by the reconstruction accuracy, Q. The reconstruction obtained by the SPGL1 using DCT on 128×128 patches can be seen in Figure 6.21(a). This also shows the same behaviour at the top and centre of the signal. The FK domain of the SPGL1 reconstruction can be seen in Figure 6.21(b) with no aliasing.

The seismic signal at the top and centre of the x-t domain is very steep, with steeper dips causing problems in reconstruction. This is because we used algorithms on 128×128

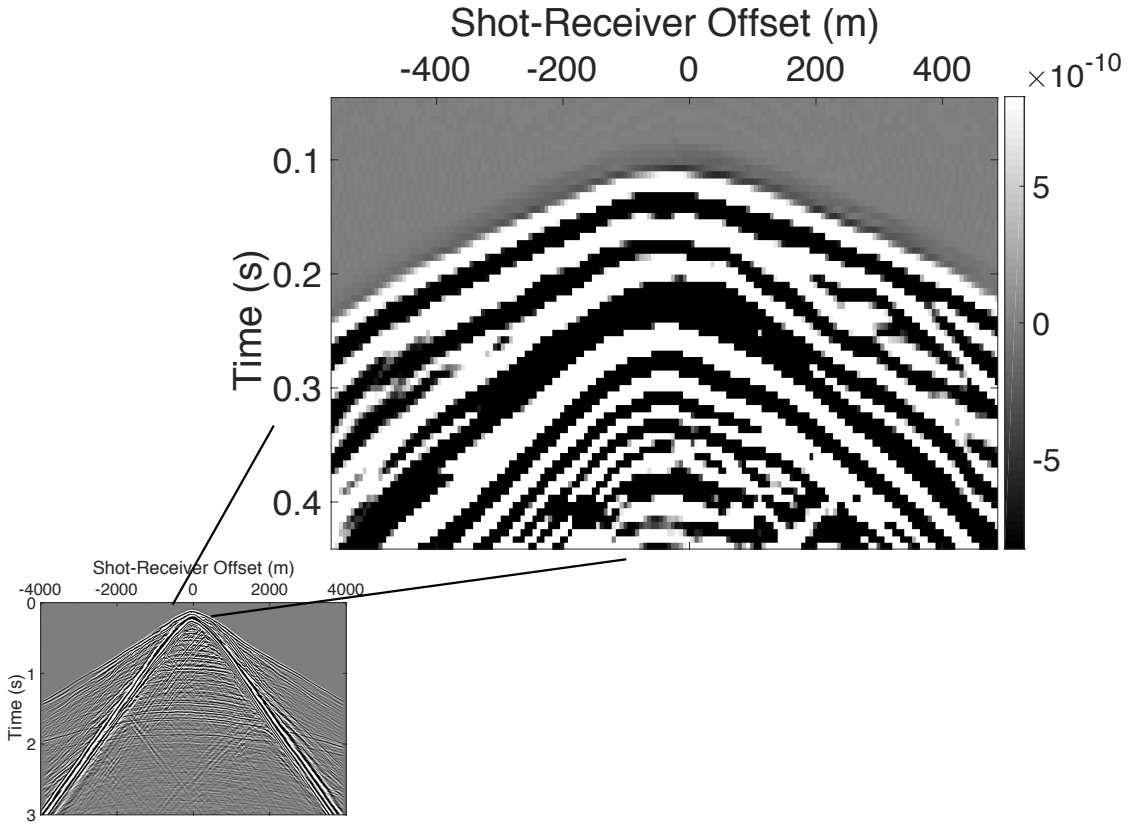


Fig. 6.16 Zoomed version of a line of receivers close to the source. It shows that the signal at the top is not zero, but very small with very fast changes.

patch sizes and the basis functions did not capture the fine details necessary to represent the signal at the top. A zoomed version of the top centre of a line of receivers close to the source is shown in Figure 6.16. We can see that the signal is not zero there but very small, with fast changes. Working on smaller patch sizes can help since the algorithms can focus more on reconstructing smaller structures. Thus, we repeat the experiments using algorithms on 8×8 patches and reconstruct the signals using POCS, SPGL1 with DCT bases, RVM with DCT bases and the BPFA all on 8×8 patches. We then use the learned bases from the BPFA with the SPGL1 and the RVM to reconstruct the signals.

Figure 6.22(a) shows the reconstruction using the BPFA. It obtains high reconstruction accuracy with very small distortion at the top centre of the signal as opposed to the reconstructions of the other algorithms. As we discussed, this is because we use 8×8 patches. The FK domain in Figure 6.22(b) does not show any signs of aliasing or noise.

The POCS reconstruction on 8×8 patches is included in Figure 6.23(a). It does not show any distortions at the top centre of the signal, nevertheless the overall reconstruction

quality is low. This is also evident in the FK domain in Figure 6.23(b) where there is strong incoherent noise. Figure 6.24(a) shows the reconstruction for the SPGL1 using the DCT bases. Again, there is no distortion at the top centre of the signal but the reconstruction quality is not high in general as seen in the FK domain in Figure 6.24(b).

Using the SPGL1 with the dictionary of bases learned by the BPFA improves the reconstruction as seen in Figure 6.25(a) with no distortions at the top centre. In addition, the overall reconstruction quality is high, with the FK domain in Figure 6.27(b) having no noise. The same behaviour can be seen with the RVM and DCT in Figure 6.26(a) obtaining the reconstruction without any distortions at the top. Improved reconstruction is obtained by using the learned dictionary of bases by BPFA in Figure 6.27(a). The improvement can also be seen in the FK domains where the RVM with DCT in Figure 6.26(b) has more incoherent noise compared to the FK domain of the RVM with the learned bases in Figure 6.27(b).

Avoiding distortions in specific region of seismic signals

By using the patch size of 8×8 we were able to obtain reconstructions without any distortions at the top centre of the signals. This is different to the ones in Figures 6.19 - 6.21 where we used the 128×128 versions of three algorithms and had distortions. Nevertheless, the overall reconstruction quality is better when using the 128×128 versions except the top part. Thus, we need to separate the seismic signals closer to the source in regions where different algorithms can operate. For example, at the top of the signal, an algorithm that operates on 8×8 patches should be used and then another algorithm on 128×128 can reconstruct the rest of the regions.

To identify which algorithm works best in each region, we calculate the mean reconstruction accuracy, Q, over all 128 receiver lines for the three percentages of receivers used in three different regions. From Figure 6.16, we can see that the top centre of the signal can be approximately obtained from the first 30 time samples (0.18/0.006 seconds). Table 6.2 shows the mean Q over all receiver lines for only the first 30 samples. Negative values for Q translate to reconstruction accuracy that is bad. The ratio between the norm of the original and the norm of the difference between the original and the reconstruction gives a fraction resulting in a negative exponent (refer to the definition of Q in equation 4.32).

We can see that the RVM using the DCT bases on 8×8 patches gives the best reconstruction accuracy. Using learned bases from the BPFA does not improve the RVM's performance since the bases learned do not capture the characteristics in this region. This is evident from the performance of the BPFA as well which is poor. The same

applies for the SPGL1's performance using the learned bases on 8×8 patches. Using the DCT bases by the SPGL1 on 8×8 patches gives good results but not as good as the RVM's. The POCS results on 8×8 patches are in general poor but still better than the POCS reconstructions on 128×128 since the larger patch size does not help in this region. The same applies to the SPGL1 with DCT and the RVM with DCT on 128×128 with lower reconstruction accuracy compared to their respective 8×8 configurations.

Reconstructions at other regions of the signals

We continue the evaluation in other regions as well. We split the regions in three. The first region is the top part from $1-30$ as discussed. The second region is between $31-200$ which contains a mixture of steep and smooth signals. The results are summarised in Table 6.3. We can see that the RVM using the DCT bases and operating on 128×128 obtains the best reconstruction accuracy in two percentages and the SPGL1 using the DCT bases on 128×128 is only slightly better when using 30% of receivers. In general, the configurations operating on 128×128 are now better than those operating in 8×8. In addition, the dictionary of learned bases improves the reconstruction of the RVM and the SPGL1 since the BPFA is able to learn useful bases. The only exception is when using 30% of receivers. Finally, the third region is between $201 - 500$ and the results are summarised in Table 6.4. The RVM using the DCT and operating on 128×128 obtains the best results for two percentages and the BPFA on 8×8 performs better when using 50% of receivers. In addition, the learned bases improve the results for both the RVM and the SPGL1 since the BPFA performs much better overall.

Combining algorithms for higher overall reconstruction accuracy

Using the RVM with DCT on 8×8 patches at the top of the signal provides higher reconstruction accuracy with no distortions. Then for the rest of the regions, the RVM with DCT on 128×128 provides the best accuracy overall. This combination provides the best possible accuracy and should be used if this is the requirement. Nevertheless, if faster computational time is needed, for practical purposes, another combination might be more beneficial (refer to section 5.8 for the trade-off between time and accuracy). We will next analyse the performance of algorithms with different variance of available data.

Close to source (1-30 samples) - Reconstruction accuracy in Q [db]			
Percentage used	30%	50%	70%
RVM-DCT 8×8	**6.033**	**15.102**	**25.450**
RVM-Learned 8×8	0.720	9.009	19.950
RVM-DCT 128×128	-5.936	6.543	17.404
POCS 8×8	5.046	8.936	12.662
POCS 128×128	-10.005	-1.953	6.109
SPGL1-DCT 8×8	4.832	13.257	23.347
SPGL1-Learned 8×8	3.032	12.108	23.828
SPGL1-DCT 128×128	-1.780	6.610	20.992
BPFA 8×8	-7.673	6.153	19.396

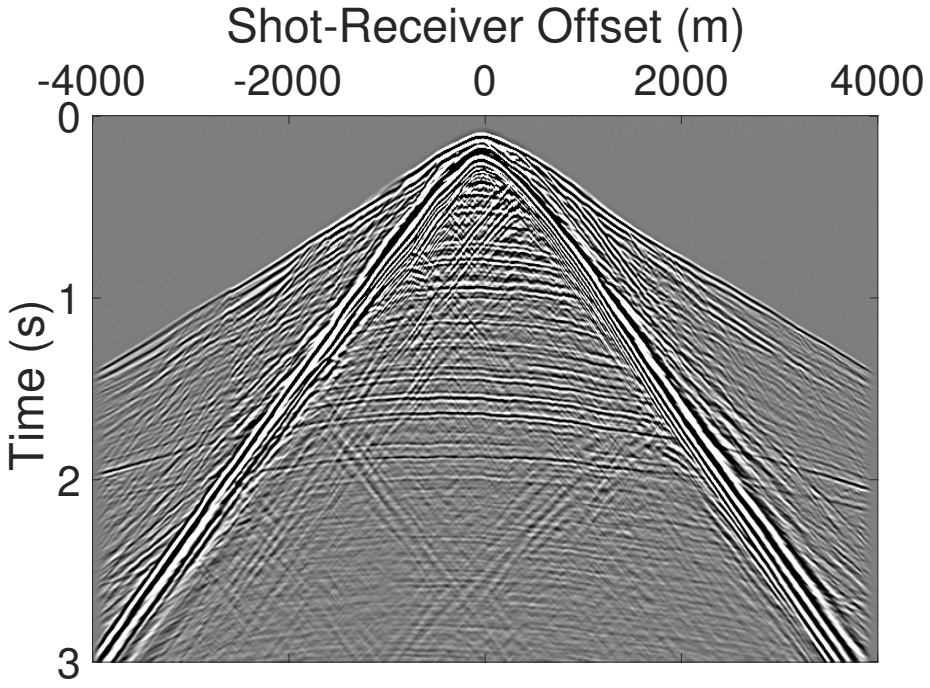
Table 6.2 Mean Q for x-t domain for close to source (1-30 time samples).

Close to source (31-200 samples) - Reconstruction accuracy in Q [db]			
Percentage used	30%	50%	70%
RVM-DCT 8×8	4.204	11.080	22.698
RVM-Learned 8×8	-0.446	14.535	28.031
RVM-DCT 128×128	6.597	**28.475**	**44.550**
POCS 8×8	4.6786	8.3327	12.220
POCS 128×128	-1.326	19.315	28.721
SPGL1-DCT 8×8	3.523	9.772	20.213
SPGL1-Learned 8×8	2.315	18.499	28.903
SPGL1-DCT 128×128	**6.604**	16.145	33.462
BPFA 8×8	-10.169	19.977	33.752

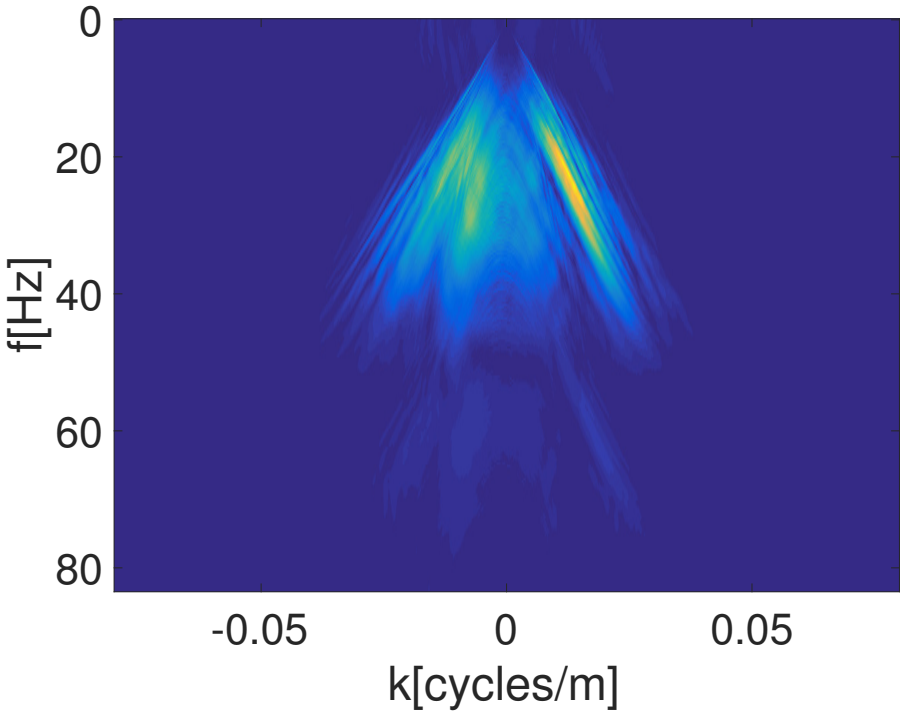
Table 6.3 Mean Q for x-t domain for close to source (31-200 time samples).

Close to source (201-500 samples) - Reconstruction accuracy in Q [db]			
Percentage used	30%	50%	70%
RVM-DCT 8×8	7.695	17.773	28.899
RVM-Learned 8×8	6.828	32.634	42.085
RVM-DCT 128×128	**21.361**	36.017	**45.782**
POCS 8×8	7.006	11.513	15.904
POCS 128×128	-3.197	23.568	29.128
SPGL1-DCT 8×8	5.803	14.726	24.972
SPGL1-Learned 8×8	11.900	28.508	35.667
SPGL1-DCT 128×128	16.825	29.574	39.459
BPFA 8×8	17.606	**36.921**	44.748

Table 6.4 Mean Q for x-t domain for close to source (201-500 time samples).

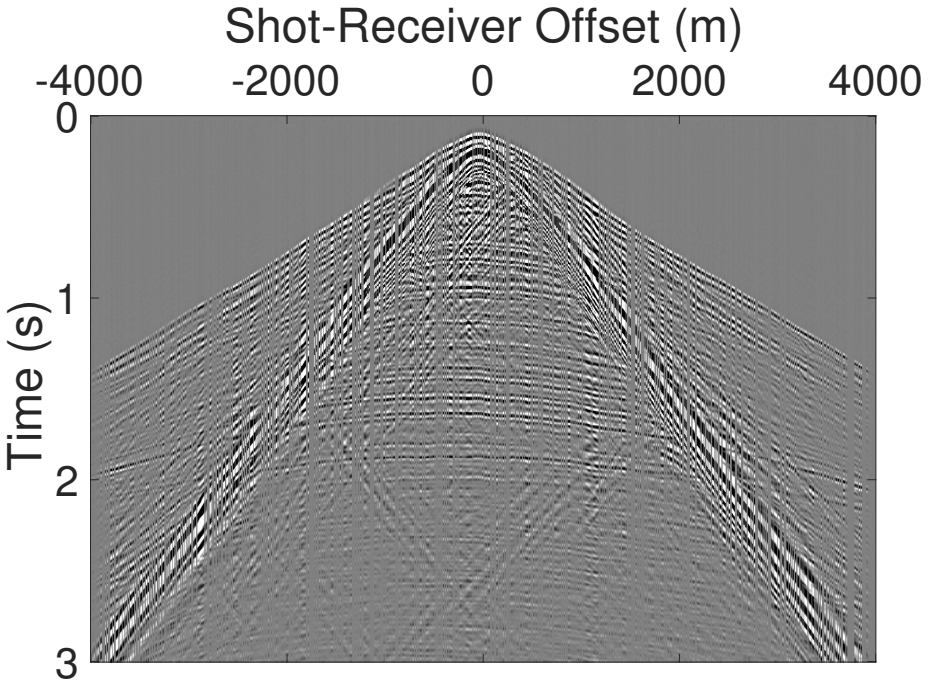


(a) Original

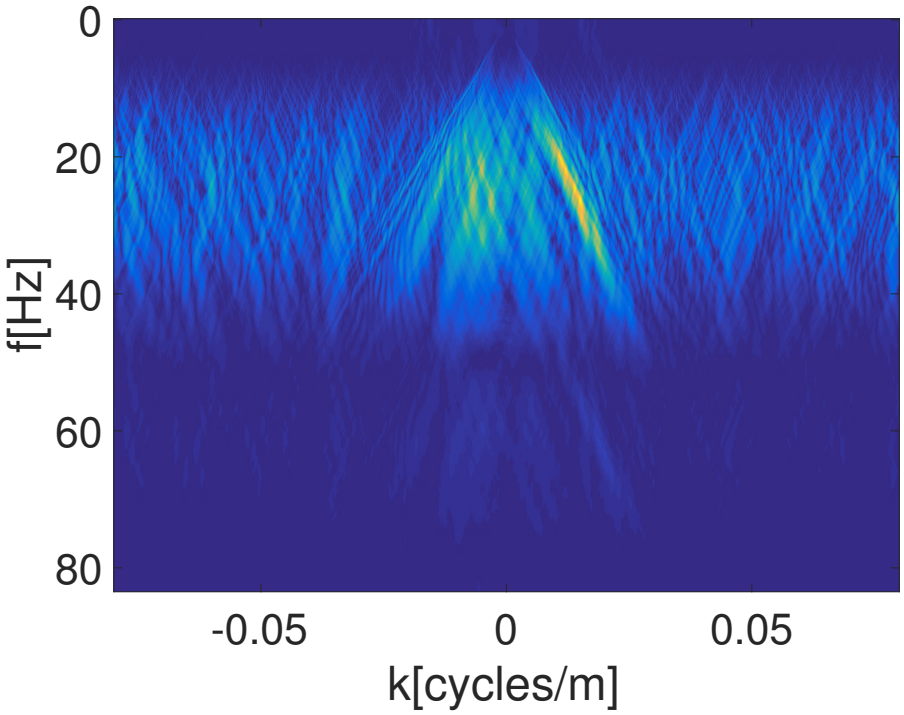


(b) FK domain of (a)

Fig. 6.17 An original receiver line (x-t domain) near the source with its respective FK domain.

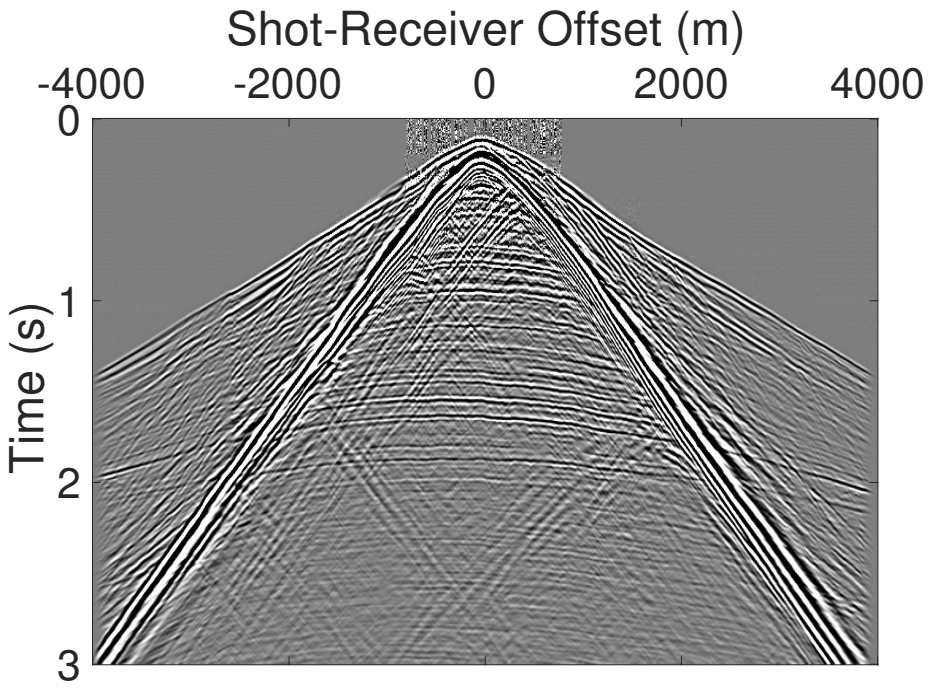


(a) Using 50% of receivers

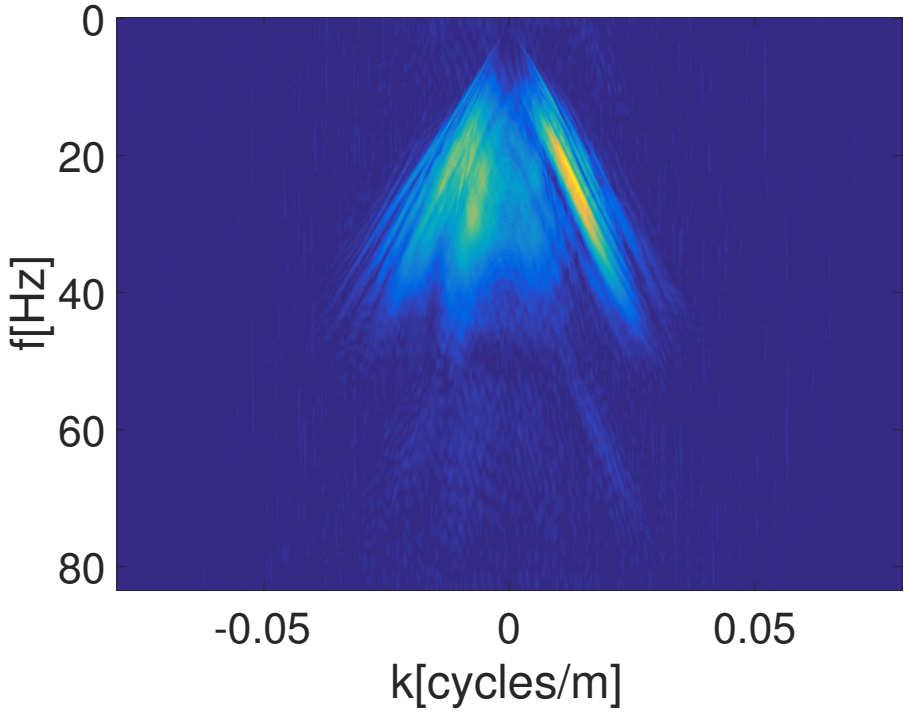


(b) FK domain of (a)

Fig. 6.18 Using 50% of receivers from a signal near the source with its respective FK domain.

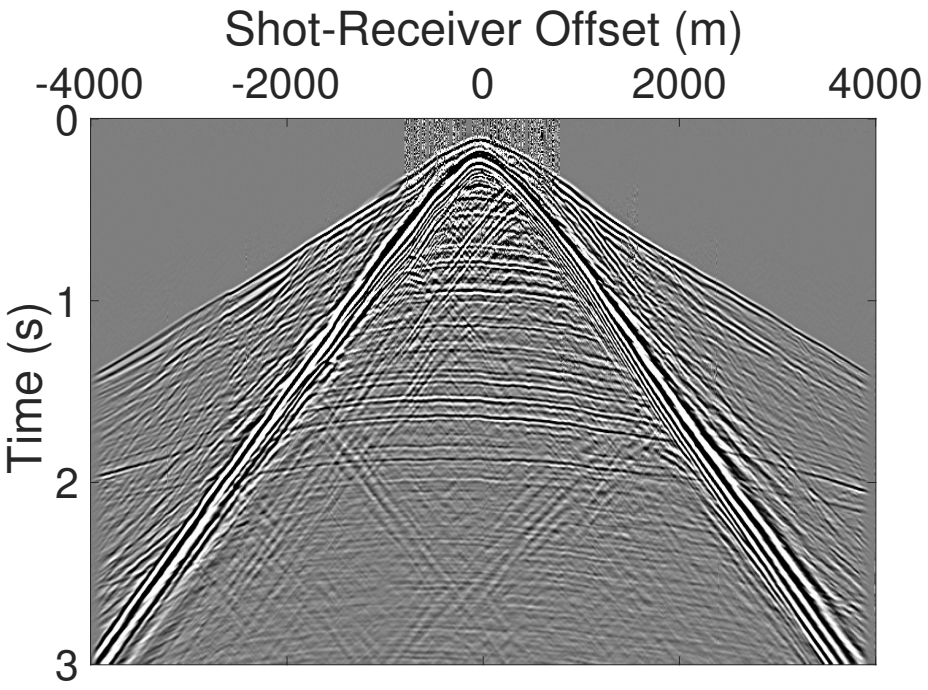


(a) RVM-DCT, 128×128 reconstruction, Q = 41.226 db

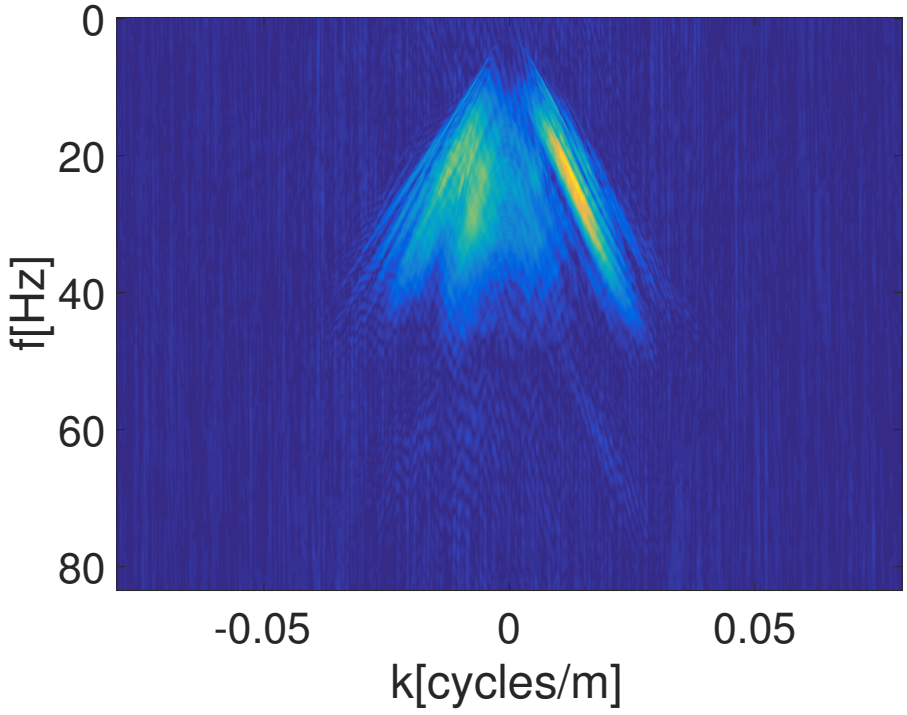


(b) FK domain of (a)

Fig. 6.19 Reconstruction using the RVM-DCT on 128×128 patches from 50% of receivers and its respective FK domain without any aliasing. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

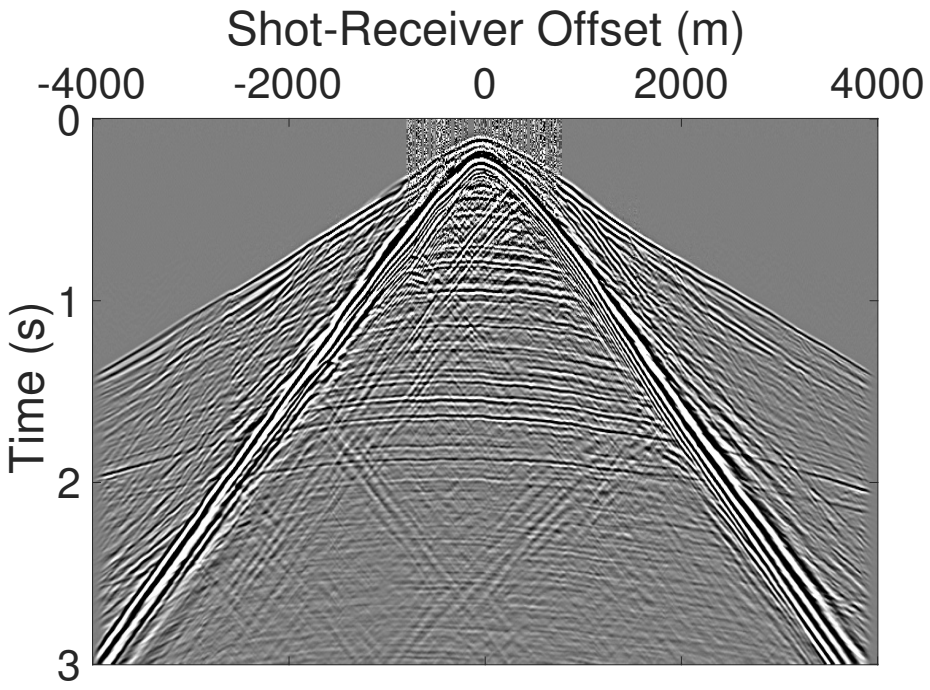


(a) POCS, 128×128 reconstruction, Q = 26.163 db

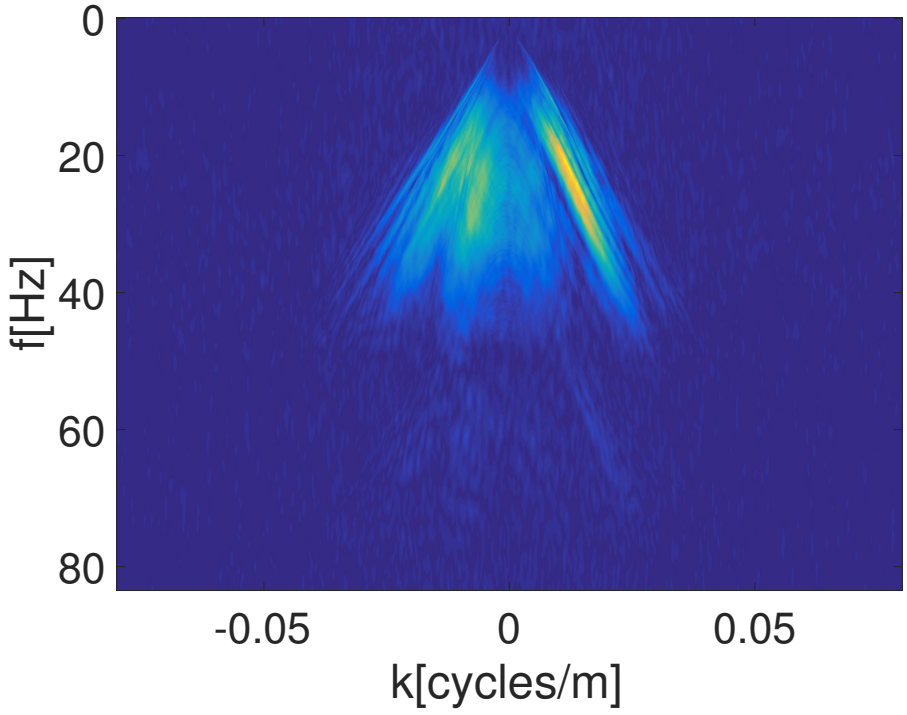


(b) FK domain of (a)

Fig. 6.20 Reconstruction using POCS on 128×128 patches from 50% of receivers and its respective FK domain without any aliasing. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

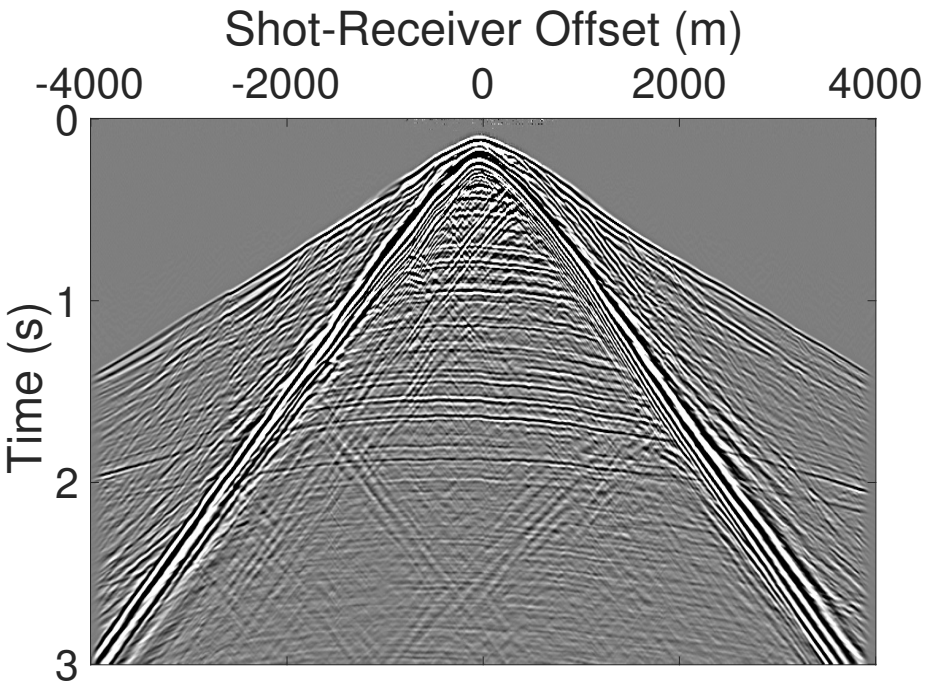


(a) SPGL1-DCT, 128 × 128 reconstruction, Q = 31.876 db

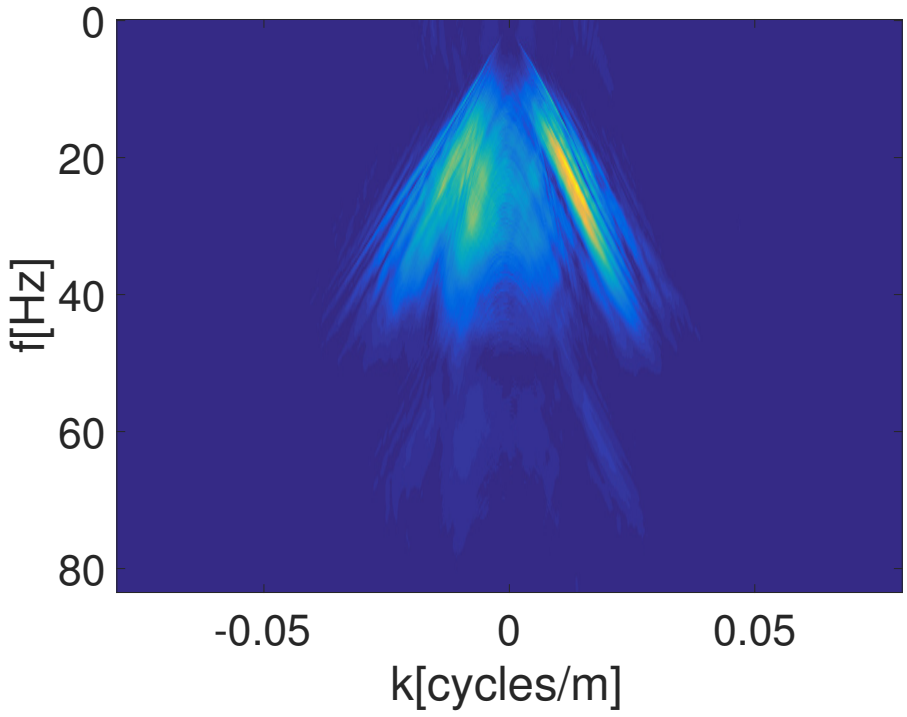


(b) FK domain of (a)

Fig. 6.21 Reconstruction using SPGL1-DCT on 128 × 128 from 50% of receivers and its respective FK domain without any aliasing. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for 201 − 500 time samples.

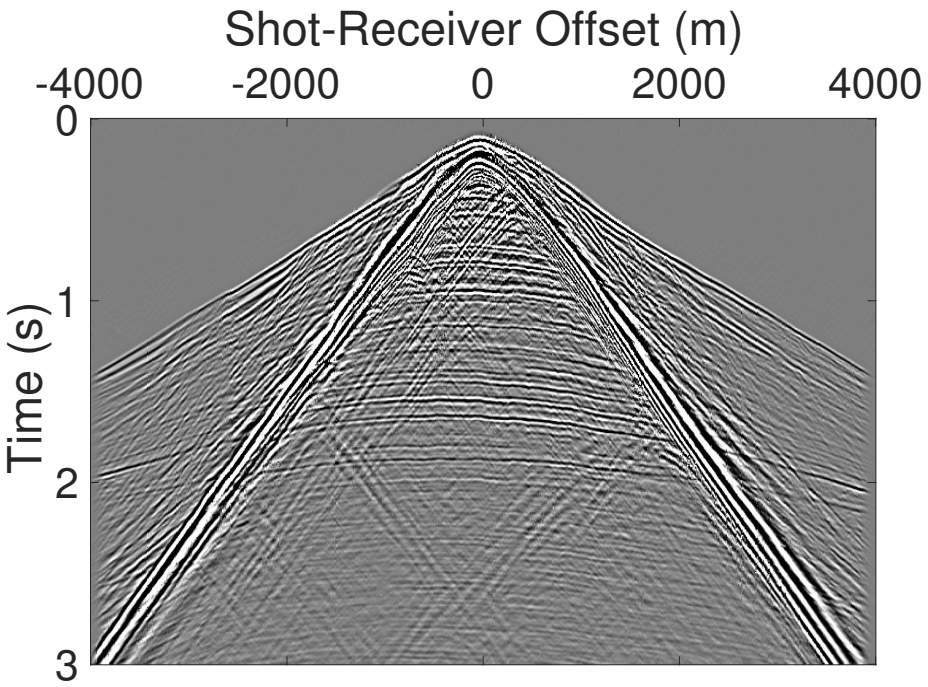


(a) BPFA, 8×8 reconstruction, Q = 42.392 db

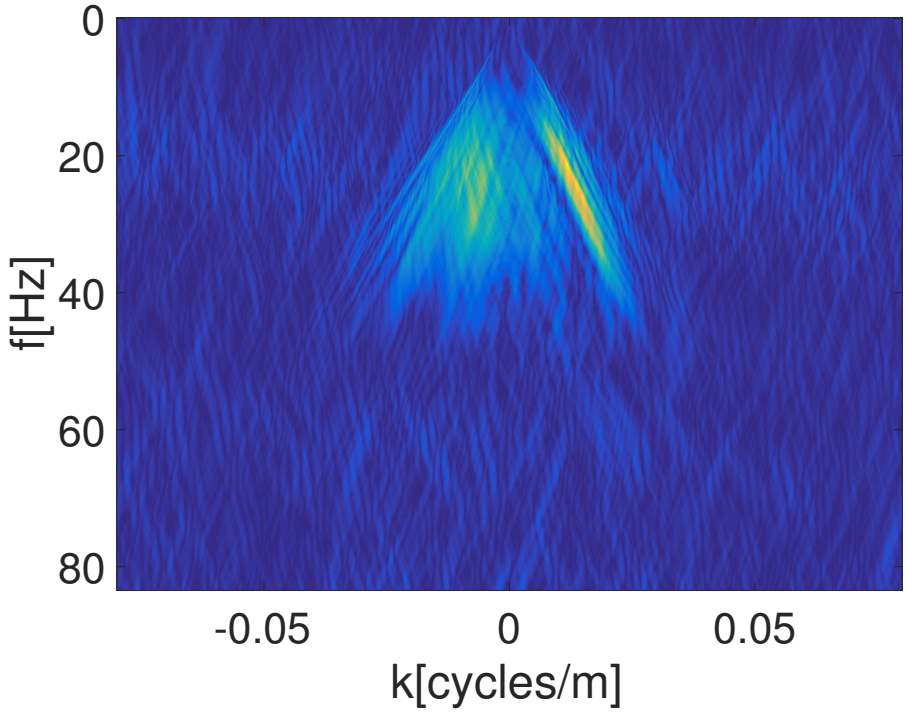


(b) FK domain of (a)

Fig. 6.22 Reconstruction using the BPFA on 8×8 patches from 50% of receivers and its respective FK domain without any aliasing or noise. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

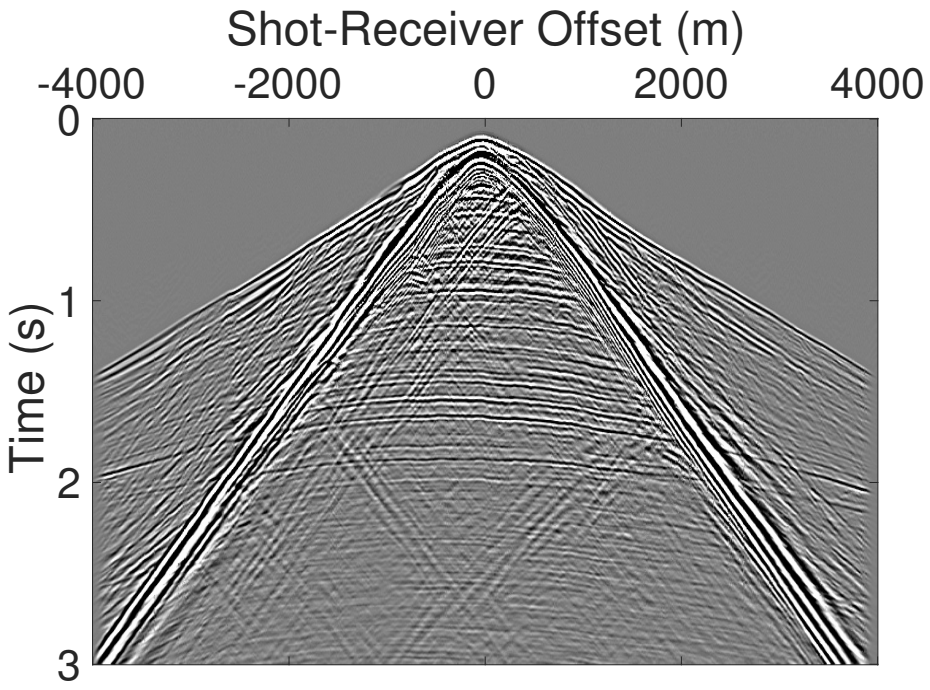


(a) POCS, 8×8 reconstruction, Q = 10.162 db

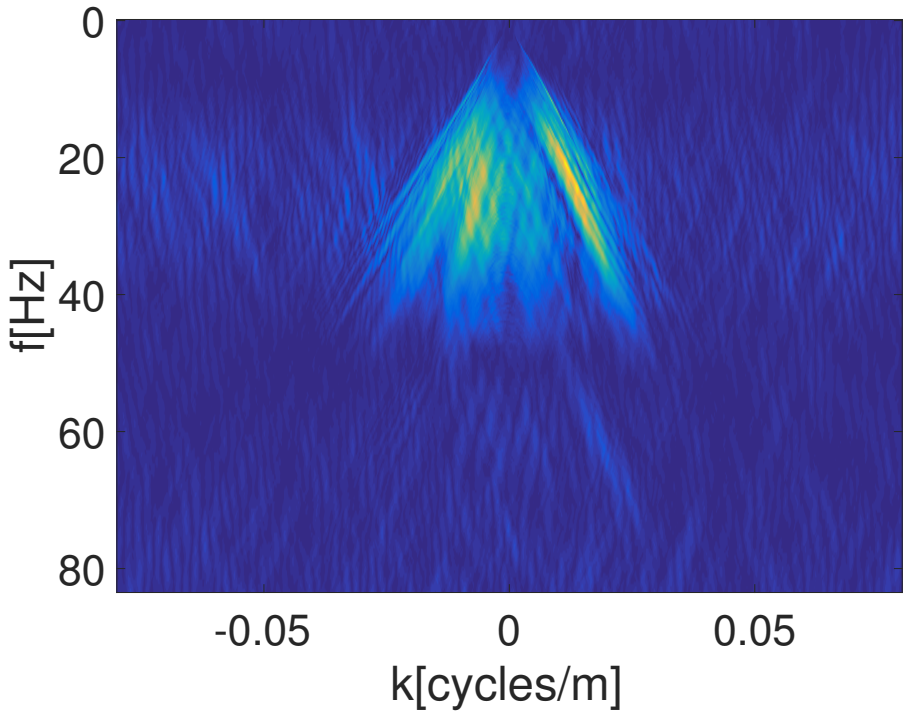


(b) FK domain of (a)

Fig. 6.23 Reconstruction using POCS on 8×8 patches from 50% of receivers and its respective FK domain with noise present. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

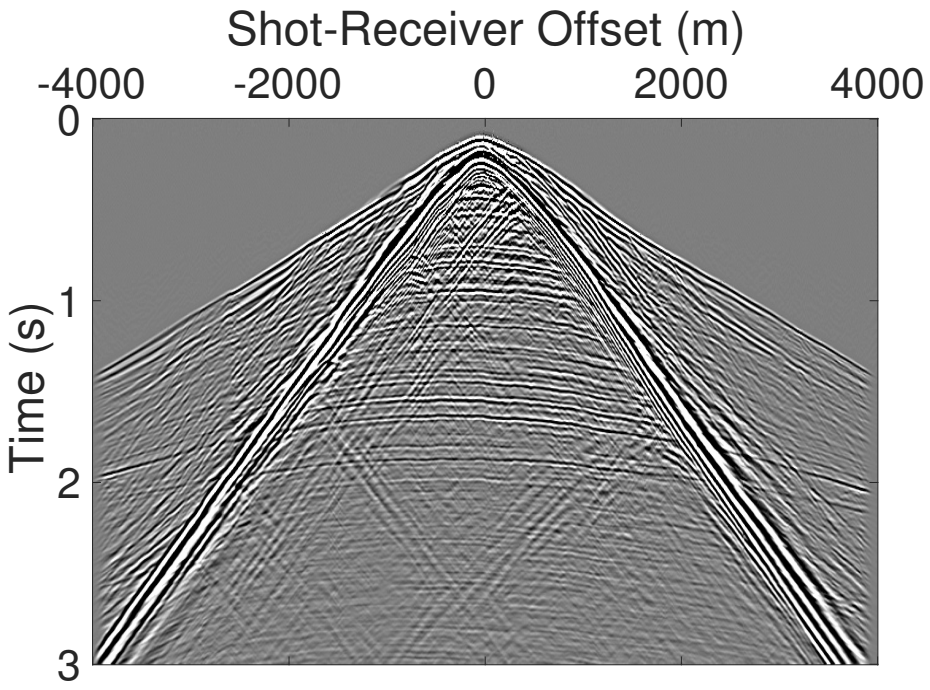


(a) SPGL1-DCT, 8×8 reconstruction, Q = 16.393 db

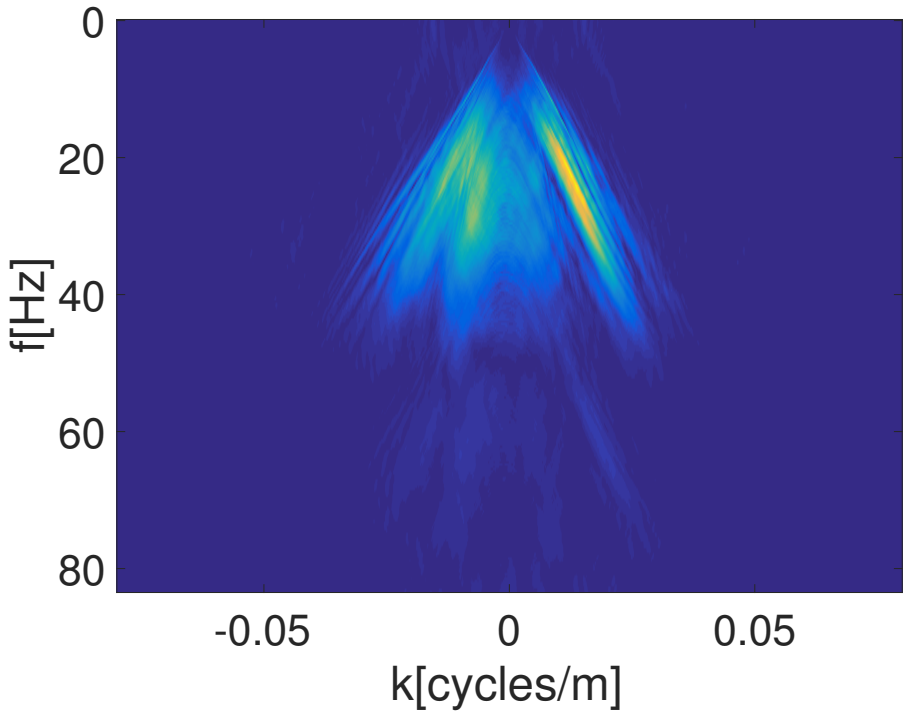


(b) FK domain of (a)

Fig. 6.24 Reconstruction using SPGL1-DCT on 8×8 patches from 50% of receivers and its respective FK domain with noise present. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

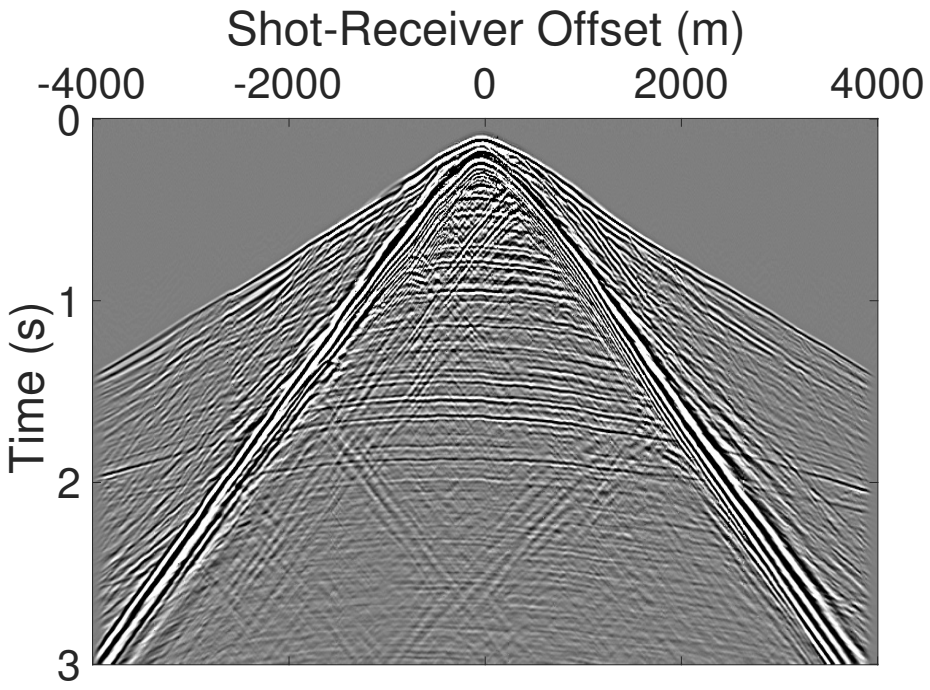


(a) SPGL1-Learned, 8×8 reconstruction, Q = 31.034 db

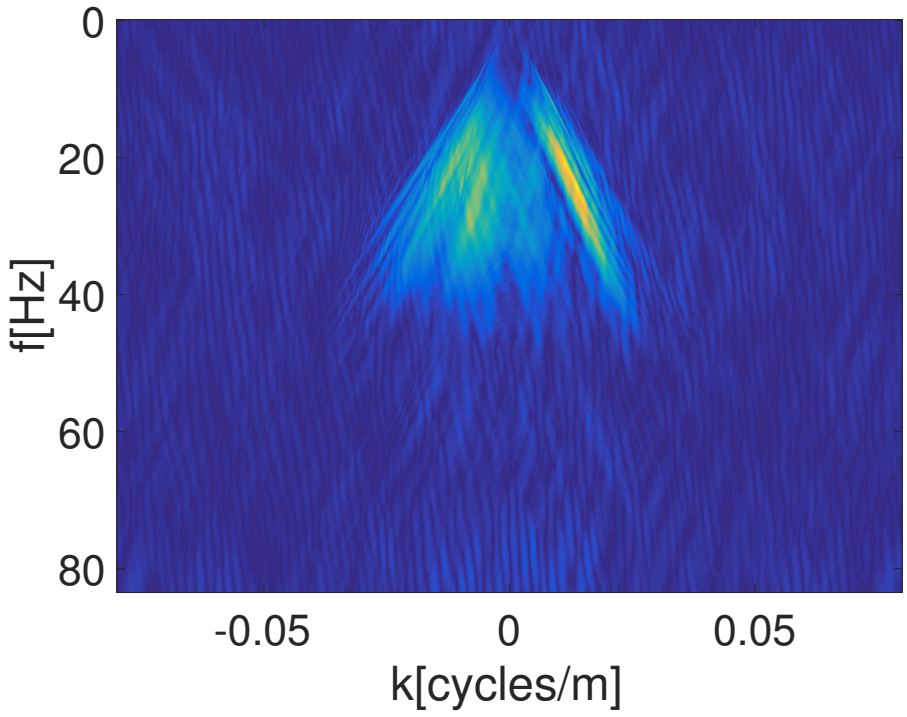


(b) FK domain of (a)

Fig. 6.25 Reconstruction using SPGL1 and learned bases from BPFA on 8×8 patches from 50% of receivers and its respective FK domain with no noise. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

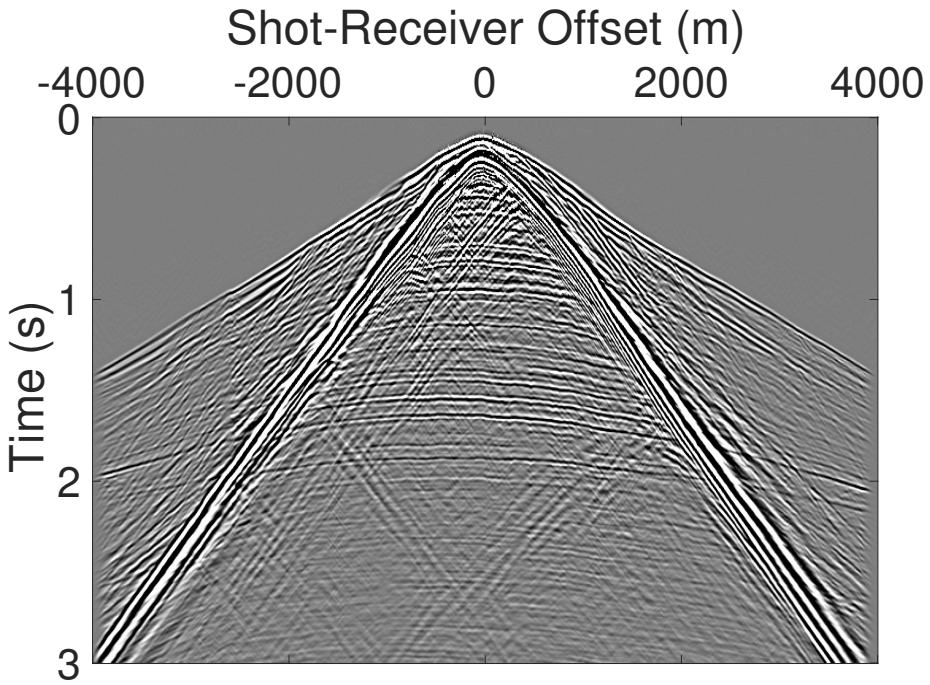


(a) RVM-DCT, 8×8 reconstruction, Q = 20.238 db

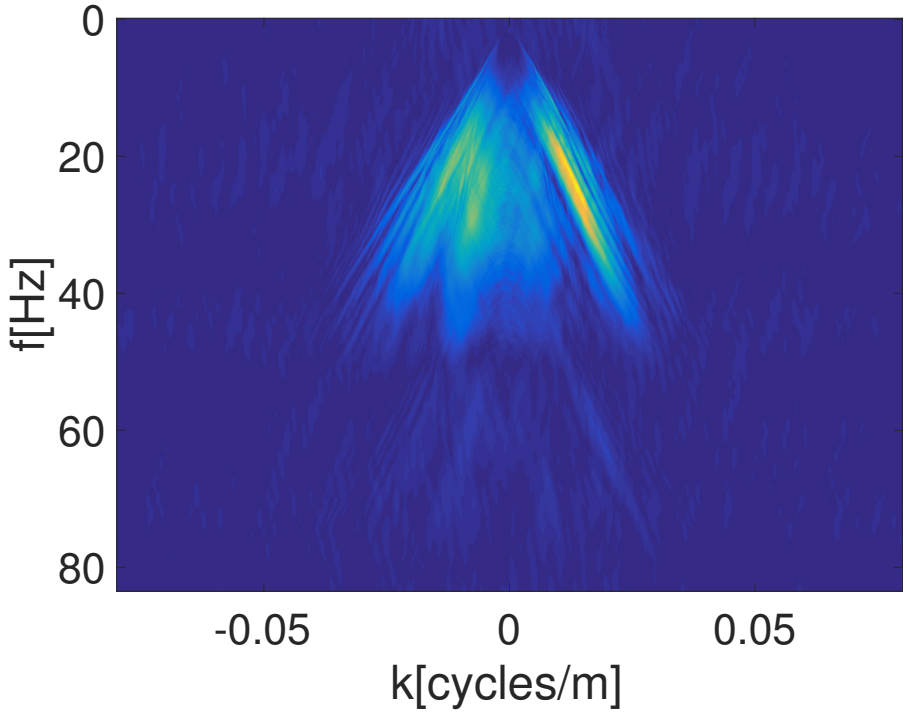


(b) FK domain of (a)

Fig. 6.26 Reconstruction using RVM-DCT on 8×8 patches from 50% of receivers and its respective FK domain with noise. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.



(a) RVM-Learned, 8×8 reconstruction, Q = 34.493 db



(b) FK domain of (a)

Fig. 6.27 Reconstruction using the RVM and learned bases from BPFA on 8×8 patches from 50% of receivers and its respective FK domain with no noise. We use the reconstruction accuracy, Q, as defined in equation 4.32. It is calculated for $201 - 500$ time samples.

6.4 Variance analysis for reconstruction accuracy

The reconstruction accuracy of any algorithm varies depending on the type of signal it processes. As we have seen in the previous section, certain regions of a seismic signal are more challenging to reconstruct than others. Depending on the curvature of the signal, how fast the changes occur, the variance of the available data, all can affect reconstruction. In order to get insight on how each algorithm behaves, we provide further analysis of the reconstruction results. We have obtained 30000 reconstructions of sections of time slices (5000 far and 5000 close to the source for three different percentages) for various algorithms. For brevity, we include only a variance analysis for 50% of receivers used. We will show, for each algorithm, a scatter plot of how the reconstruction accuracy changes depending on the variance of the available data used per section combining sections for both close and far from the source resulting in 10000 data points.

A ranked scatter plot of these variables will be more useful so as to be able to visualise the data better. We rank each variable from smallest to largest value (equal values are ranked the same) which is also a prerequisite for the Spearman's correlation coefficient,

$$\text{Spear} = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i - \mu_x)^2}\sqrt{\sum_{i=1}^{n}(y_i - \mu_y)^2}}, \tag{6.1}$$

where n are the number of sections of time slices, $\{x_i\}_{i=1}^{n}$ and $\{y_i\}_{i=1}^{n}$ are the variables of interest and μ_x, μ_y are their respective means. In this scenario, $\mathbf{x}$ contains the reconstruction accuracy, Q, for all reconstructed sections per algorithm and $\mathbf{y}$ contains the variance of the available data per section (both variables are ranked). This number varies between -1 and $+1$. Positive number means that the variables are positively correlated and as the one increases/decreases so does the other. Negative sign translates to variables being negatively correlated and as the one increases/decreases the other one behaves in the opposite way. The higher the absolute value of the Spearman's correlation coefficient the more positively/negatively correlated they are. We will use this coefficient to identify if the reconstruction accuracy is correlated with the variance of the data.

Before discussing the scatter plots, we show the variance of the available data for 50% of receivers per section. Figure 6.28 shows the variance of all sections for close to the source with the top centre region having the largest values. If we observe the reconstructions in Figures 6.19 - 6.21, we can see that the region of bad reconstructions coincides with the high variance. Note that the rest of the sections have a variance that is orders of magnitude smaller and thus not visible.

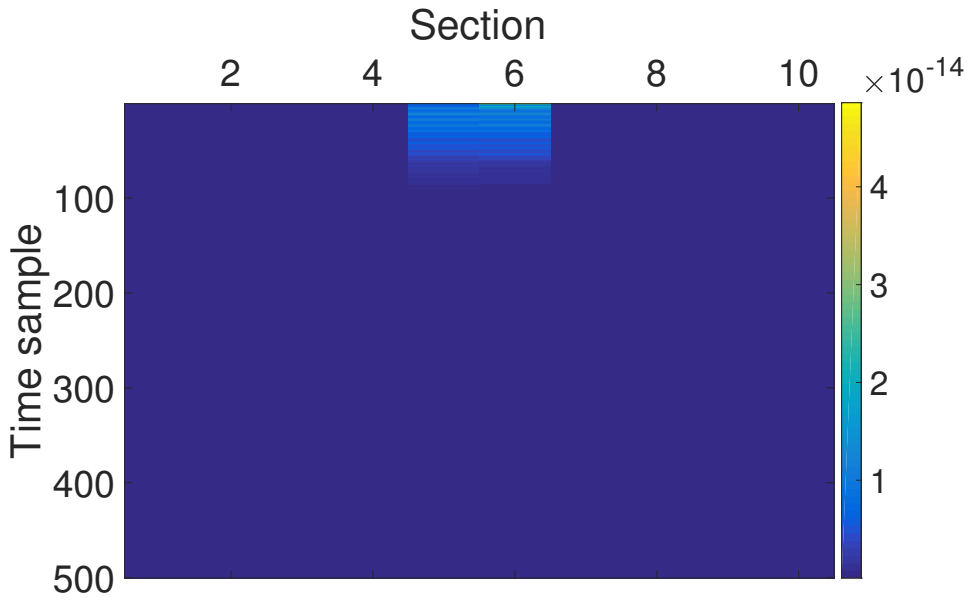


Fig. 6.28 Variance per section and per time sample for close to source.

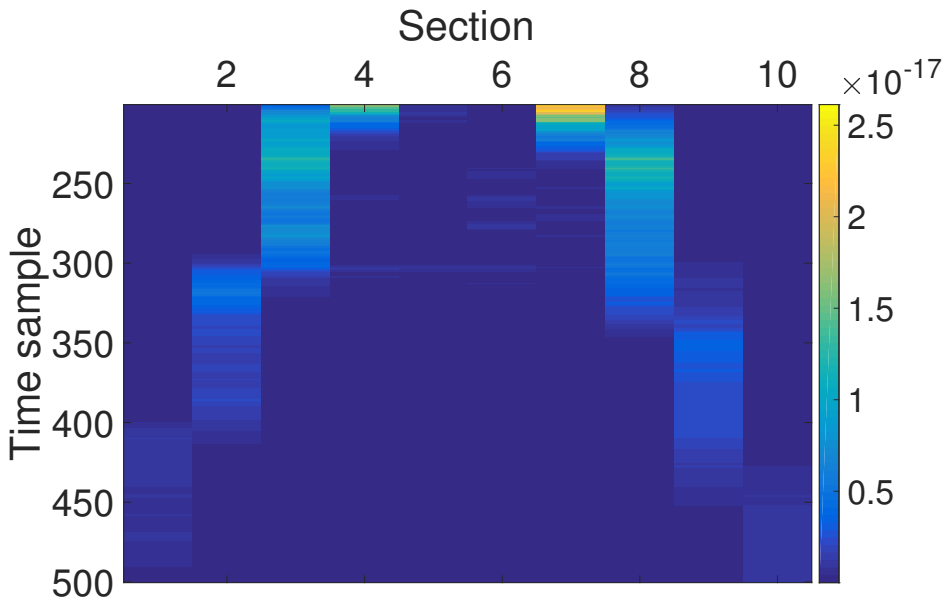


Fig. 6.29 Variance per section and per time sample for close to source after the 200-th sample.

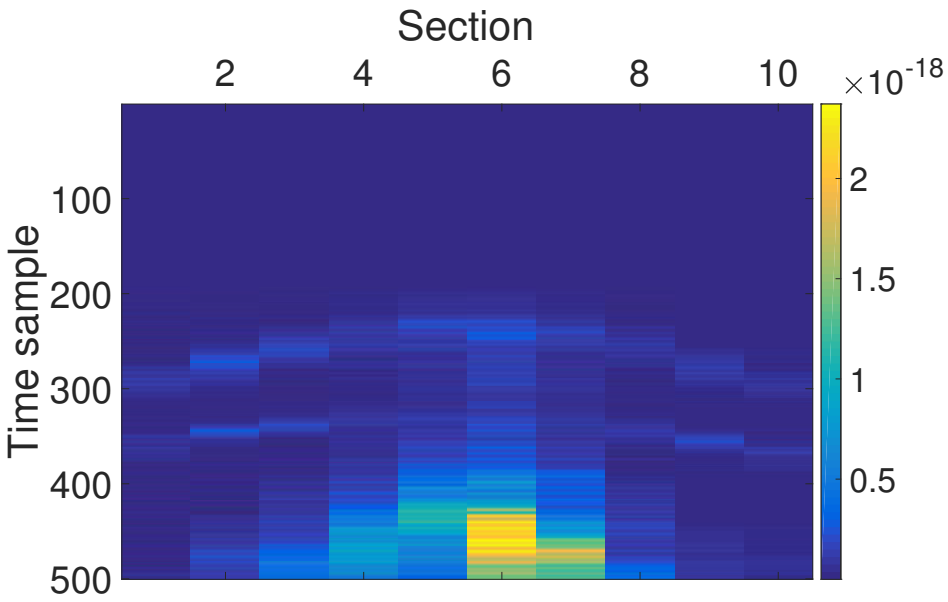


Fig. 6.30 Variance per section and per time sample for far from source.

We include the sections after the 200-th time sample for completeness in Figure 6.29. It can be seen that the variance is much smaller than the variance in Figure 6.28. For the receiver lines that are far from the source, we plot the variance in Figure 6.30. We can see that the variance is generally low until the 200-th time sample and peaks at the very end. This pattern matches with how an original receiver line looks like (refer to Figure 6.5(a)).

Scatter plots

We calculate the variance of the available data in each section and the reconstruction accuracy in Q for each algorithm. This gives 10000 pairs with many of the variances very close to zero and with different ranges. Thus, we ranked each variable before plotting as mentioned. Figure 6.31 shows the scatter plot for POCS on 8×8 patches. We can see that for sections with low variance, the reconstruction is bad as well as sections with very high variance. When the variance is in the middle, the reconstruction accuracy is spread around. The Spearman's correlation coefficient is positive at 0.6418 meaning that as the variance increases so does the reconstruction. Figure 6.32 shows a wider scatter plot. This is reflected in the correlation coefficient at 0.6106 showing positive correlation but less than the 8×8 version. This means that the reconstruction of POCS on 128×128 is not as correlated with the variance of the data.

Figure 6.33 shows the scatter plot of SPGL1 using DCT on 8×8 patches. This is similar to POCS on 8×8. If we use the learned dictionary of bases with the SPGL1, we can see in Figure 6.34 that the scatter plot is more spread. Moving on to the SPGL1 with DCT bases on 128×128, the scatter plot in Figure 6.35 shows the higher the variance the better reconstruction accuracy but too high variance results in poor reconstruction. The Spearman's correlation coefficient is 0.7308. The BPFA's scatter plot on 8×8 is given in Figure 6.36. This is less correlated with the Spearman's correlation coefficient at 0.6382. The most correlated results are given by the RVM using DCT on 128×128 patch size with the Spearman's correlation coefficient at 0.8209. At smaller patch sizes, the scatter plot with DCT bases is given in Figure 6.38 and with learned bases in Figure 6.39. We can see that the performance of algorithms varies and care is needed. The RVM with DCT bases on 128×128 obtains the best performance in general. Nevertheless, too much variance is bad as we have seen in the top centre signal of the close to source lines. In this case, the RVM on 8×8 patches using the DCT is more suitable. The learned dictionary of bases is not suitable since the BPFA does not learn useful bases at such high frequencies. Thus, different algorithms produce different predictions with varying degrees of accuracy. An uncertainty associated with each prediction would be desirable and this will be the subject of the next chapter.

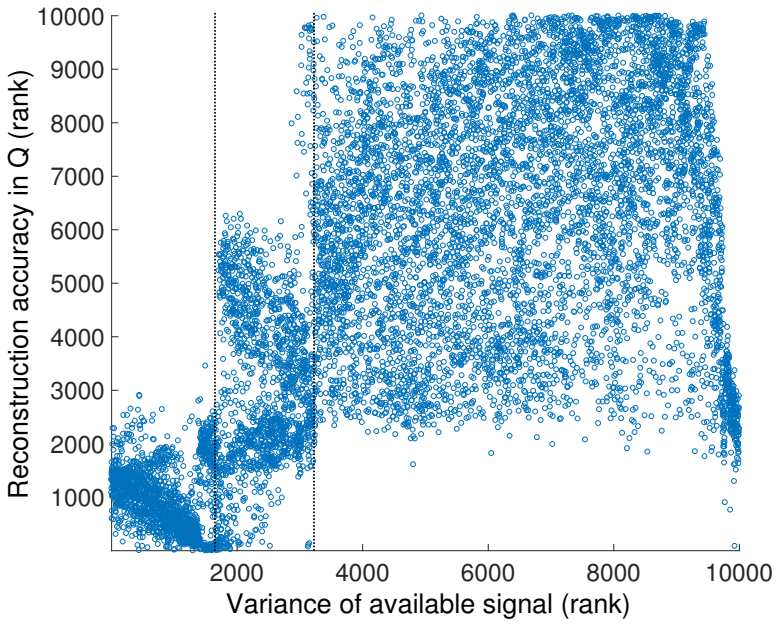


Fig. 6.31 Variance analysis for POCS, 8×8 with Spear = 0.6418. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

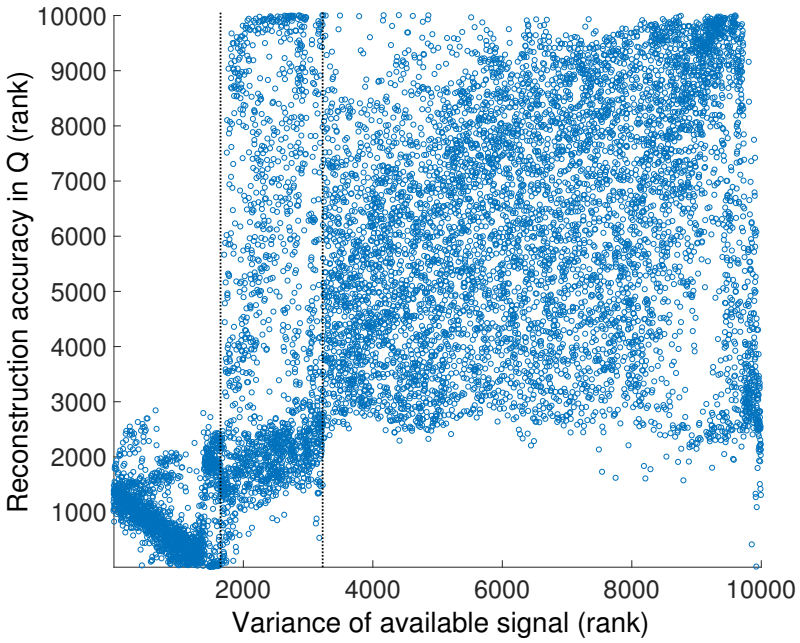


Fig. 6.32 Variance analysis for POCS, 128×128 with Spear $= 0.6106$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

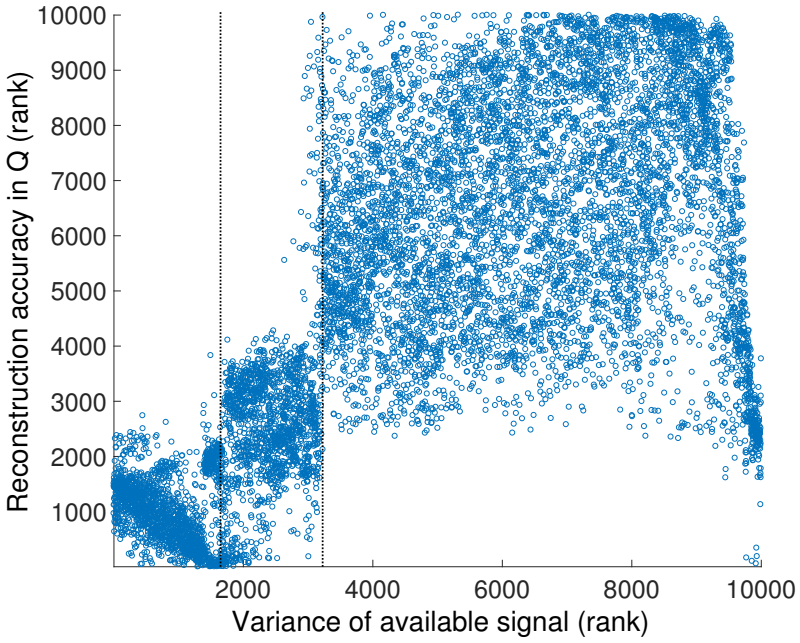


Fig. 6.33 Variance analysis for SPGL1-DCT, 8×8 with Spear $= 0.6811$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

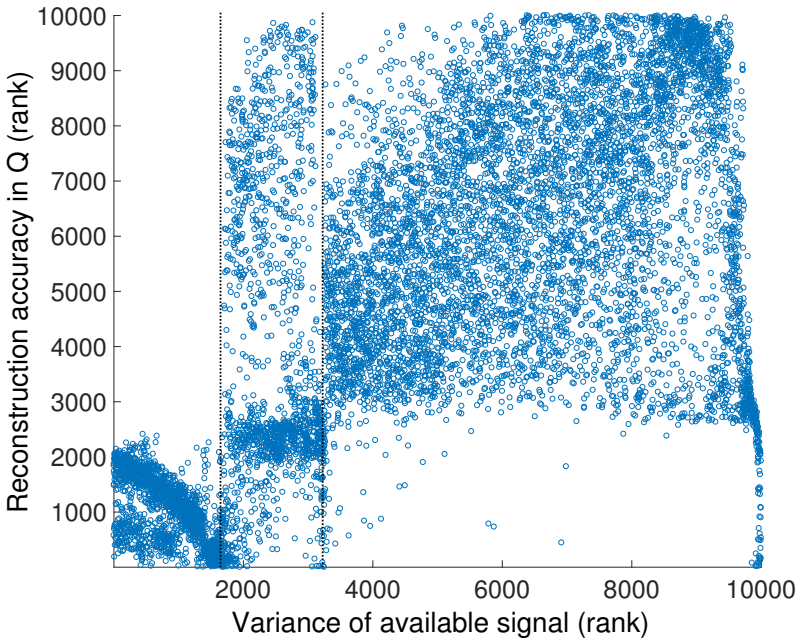


Fig. 6.34 Variance analysis for SPGL1-Learned, 8×8 with Spear $= 0.6225$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

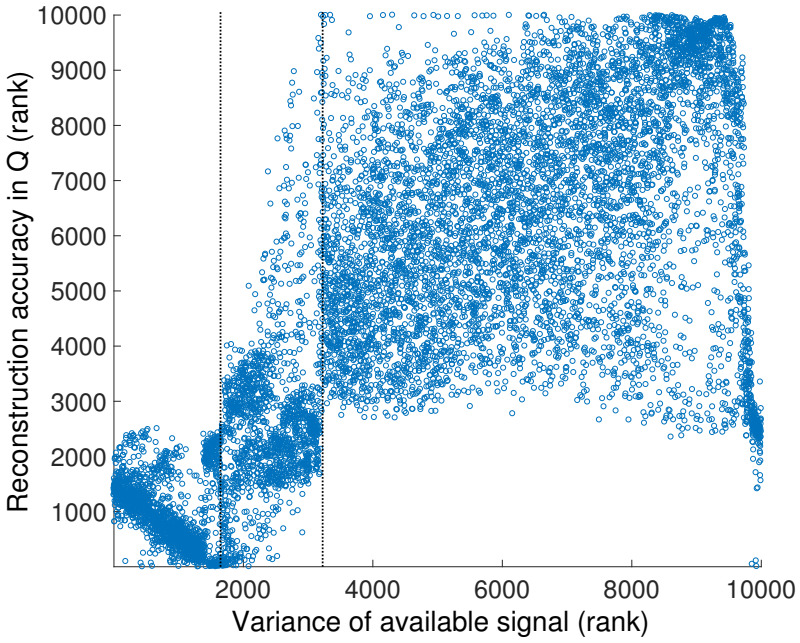


Fig. 6.35 Variance analysis for SPGL1-DCT, 128×128 with Spear $= 0.7308$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

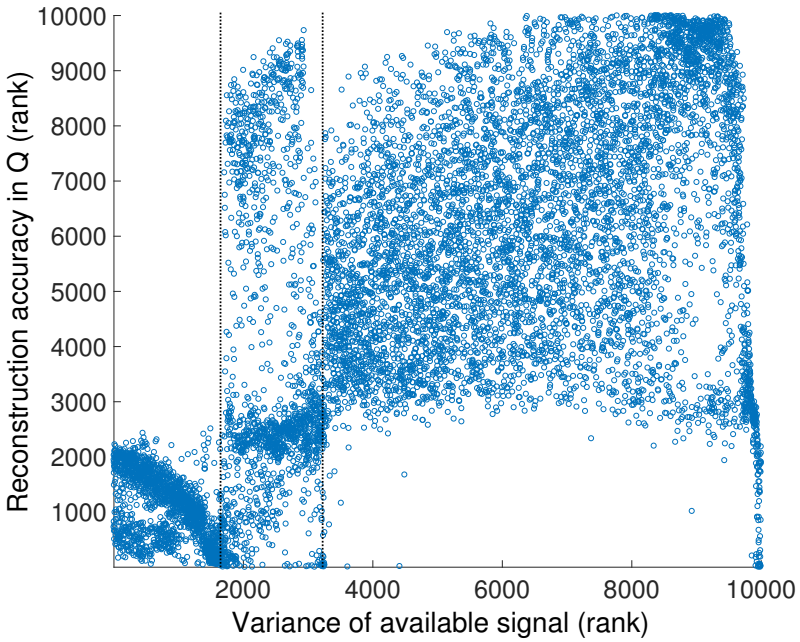


Fig. 6.36 Variance analysis for BPFA, 8×8 with Spear $= 0.6382$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

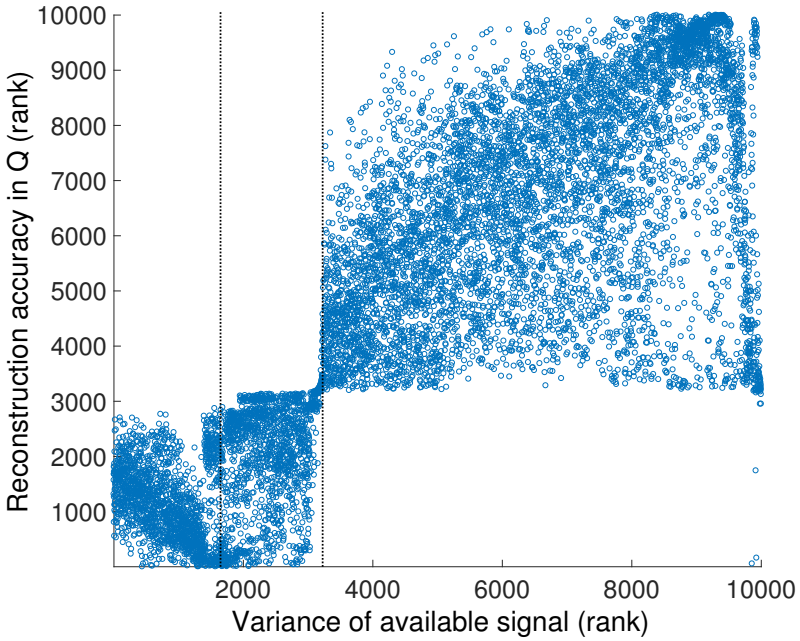


Fig. 6.37 Variance analysis for RVM-DCT, 128×128 with Spear $= 0.8209$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

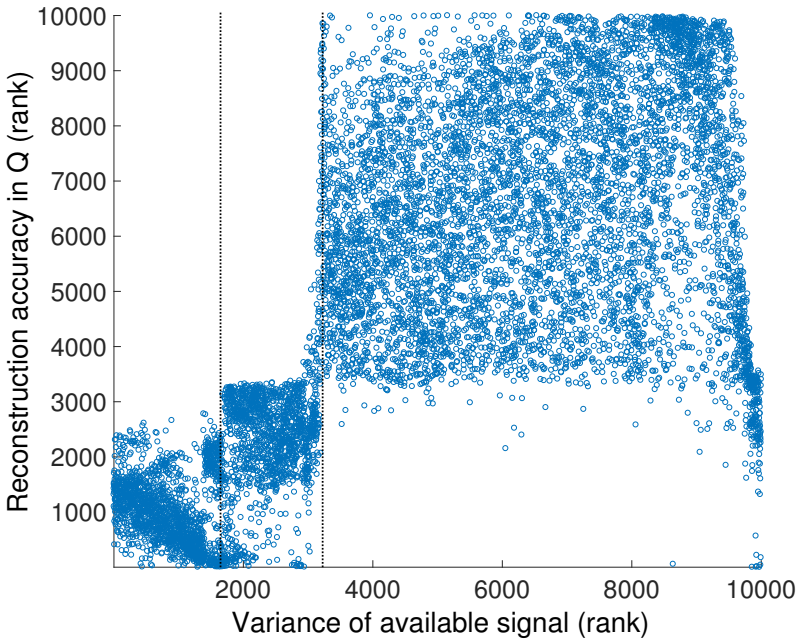


Fig. 6.38 Variance analysis for RVM-DCT, 8×8 with Spear $= 0.6954$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

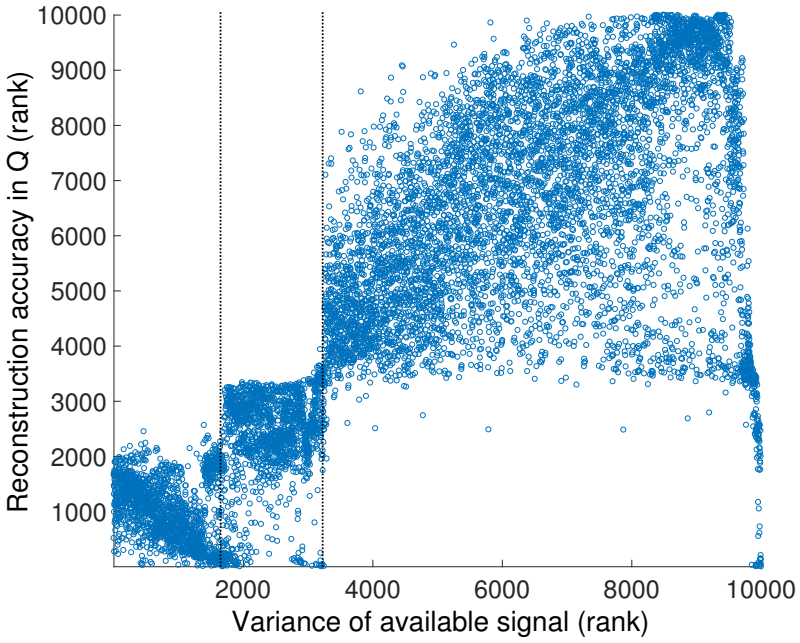


Fig. 6.39 Variance analysis for RVM-Learned, 8×8 with Spear $= 0.7867$. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

Chapter 7

Uncertainty Quantification for Seismic Compressive Sensing

Seismic Compressive Sensing (CS) is used in order to improve seismic data acquisition and survey design. Nevertheless, most methods are ad hoc and their only aim is to fill in the gaps in the data. Algorithms might be able to predict missing receivers' values, however it is also desirable to be able to associate each prediction with a degree of uncertainty. We propose to use the Bayesian statistics framework in CS to achieve this. In previous chapters, we proposed to use the Relevance Vector Machine (RVM) and the Beta Process Factor Analysis (BPFA) for predictions. In this chapter we will use the same techniques to create uncertainty maps, associating a level of confidence with each prediction. We will use the BPFA and the variance of its Gibbs samples to achieve this as well as use the RVM's predictive variance and modifications. We will first describe the modifications of the RVM along with a short review of how we can obtain uncertainty maps using the BPFA. Then we will provide results on individual sections of time slices as well as provide a quantitative comparison between uncertainty maps.

7.1 Relevance Vector Machines and modifications

In chapter 4, we have seen that the RVM infers a predictive distribution for $t^{(*)}$ given by $\mathcal{N}(m_*, \sigma_*^2)$ where m_* is the predictive mean and σ_*^2 is the predictive variance. We have used m_* as the value for our prediction as seen in equation 4.30 and σ_*^2 as seen in 4.31. This predictive distribution is heavily dependent on the model, since it depends on $\boldsymbol{\phi}(\mathbf{k}^{(*)})$ which are the basis functions evaluated at $\mathbf{k}^{(*)}$. It is customary to choose basis functions for the dictionary which decay quickly when moving away from their centre, or

basis functions with finite, compact support such as the Haar wavelets. Therefore the degenerate case is possible, that is, $\boldsymbol{\phi}(\mathbf{k}^{(*)})$ is close to, or even equal to zero, and thus the predictive probability distribution becomes $\mathcal{N}(0, \sigma^2)$ which is meaningless (we used this property to create a cascade of RVMs in section 4.3). Furthermore, it was noted by Rasmussen and Quiñonero Candela (2005) that the RVM produces predictive variances opposite to what would be desirable (i.e. small variance close to the training data and large variance away from it). We thus use the Discrete Cosine Transform (DCT) as dictionary of basis functions with the RVM to minimise the problem of degeneration and at the same time obtain higher reconstruction accuracy as discussed in subsection 4.4.3. Before illustrating the uncertainty maps produced, we describe two modifications for the uncertainty of the RVM.

7.1.1 Healing the RVM with augmentation

Rasmussen and Quiñonero Candela (2005) proposed to augment the RVM model by adding a basis function centred at a test point (missing receiver) which might potentially lie far from the support of all the previously added basis functions. By doing that, the training of the model does not change before test time. Thus, for every missing/test data point, one new basis function is added which is centred at $\mathbf{k}^{(*)}$. By adding the new basis function, the posterior distribution of the model weights has changed to

$$\boldsymbol{\mu}_* = \sigma^{-2}\boldsymbol{\Sigma}_* \begin{bmatrix} \boldsymbol{\Phi}^T \\ \boldsymbol{\phi}_*^T \end{bmatrix} \mathbf{t}, \tag{7.1}$$

$$\boldsymbol{\Sigma}_* = \begin{bmatrix} \boldsymbol{\Sigma}^{-1} & \sigma^{-2}\boldsymbol{\Phi}^T\boldsymbol{\phi}_* \\ \sigma^{-2}\boldsymbol{\phi}_*^T\boldsymbol{\Phi} & \alpha_* + \sigma^{-2}\boldsymbol{\phi}_*^T\boldsymbol{\phi}_* \end{bmatrix}^{-1}, \tag{7.2}$$

where $\boldsymbol{\Sigma}$ is the covariance of the posterior distribution of the original RVM and $\boldsymbol{\phi}_*$ is the added basis function centred at the unknown data point but calculated for all data points, augmenting the model.

We can then use this augmented posterior distribution to obtain the augmented predictive mean, $m_*(\mathbf{k}^{(*)})$ and predictive variance, $v_*(\mathbf{k}^{(*)})$ of the model defined by

$$m_*(\mathbf{k}^{(*)}) = m_* + \frac{e_* q_*}{\alpha_* + s_*}, \tag{7.3}$$

and

$$v_*(\mathbf{k}^{(*)}) = \sigma_*^2 + \frac{e_*^2}{\alpha_* + s_*}. \tag{7.4}$$

The expressions for m_* and σ_*^2 are given in 4.30 and 4.31 respectively and the other variables are defined as

$$q_* = \boldsymbol{\phi}_*^T(\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{\mu})\sigma^{-2}, \tag{7.5}$$

$$e_* = \boldsymbol{\phi}_*(\mathbf{k}^{(*)}) - \sigma^{-2}\boldsymbol{\phi}(\mathbf{k}^{(*)})\boldsymbol{\Sigma}\boldsymbol{\Phi}^T\boldsymbol{\phi}_*, \tag{7.6}$$

$$s_* = \boldsymbol{\phi}_*^T(\sigma^2\mathbf{I} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}\boldsymbol{\phi}_*. \tag{7.7}$$

q_* is the quality factor and s_* is the sparsity factor as used by Faul and Tipping (2001). $\boldsymbol{\phi}_*$ is a column vector that contains the response of the new basis function at all training inputs. The variable, e_* is a new concept. It captures the error between how the current model describes the new basis function at $\mathbf{k}^{(*)}$. Furthermore, for each new basis function, the corresponding weight has to be updated which has a prior distribution $p(w_*) \sim \mathcal{N}(0, \alpha_*^{-1})$. Thus, it is necessary to set a value for α_*^{-1} which as suggested by Rasmussen and Quiñonero Candela (2005) could be set to the empirical variance of the available training data.

A popular choice for localised basis functions are the squared exponential radial basis functions that we have seen in chapter 2 in Figure 2.17. Tuning for this is necessary or to be inferred from the data as suggested by Tipping (2001). We will use this to illustrate a possible uncertainty map obtained by augmentation. However, by using augmentation the model is not truly sparse any more. We will describe a different approach in the next section.

7.1.2 RVM's change in model likelihood

Another approach of obtaining an uncertainty map is to calculate further statistics for the signal of interest. To do this we will first address the challenge of new data arriving in the model. That is, we will assume that we have trained the model of the RVM with the available receivers and then treat the missing receivers as new data points arriving. Following the approach by Faul and Tipping (2001) and Faul and Pilikos (2016) we calculate the change in the logarithm of the marginal likelihood for the current model, when a data sample $(\mathbf{k}^{(*)}, t^{(*)})$ is added. The change in the logarithm of the marginal likelihood is given by Faul and Pilikos (2016) as,

$$\Delta\mathcal{L} = -\frac{1}{2}\left[\ln 2\pi + \ln\sigma_*^2 + \left(\frac{m_* - t^{(*)}}{\sigma_*^2}\right)^2\right] \tag{7.8}$$

$$= \ln\frac{1}{\sqrt{2\pi}\sigma_*}\exp\left(-\frac{(m_* - t^{(*)})^2}{2\sigma_*^2}\right). \tag{7.9}$$

Hence the change is the logarithm of the likelihood of the new data value $t^{(*)}$ at $\boldsymbol{\phi}(\mathbf{k}^{(*)})$ given the predictive probability distribution $\mathcal{N}(m_*, \sigma_*^2)$.

Since $\sigma_* \geq \sigma$, the change lies between $-\infty$ and $\log \frac{1}{\sqrt{2\pi\sigma}}$, it can be positive. In this case the new sample affirms the model. If the likelihood of the data is small, the marginal likelihood is reduced, indicating that the model should be updated. To do so, all quantities need to be updated. Efficient update formulae can be found in Faul and Pilikos (2016).

Proposed measure based on the change

Using the above framework, we can estimate the change in likelihood when a new receiver is used in the model. The value, $t^{(*)}$, of receiver $\mathbf{k}^{(*)}$ is unknown and thus we estimate a probability distribution for its value. Let $\mathcal{S}$ be a subset of the receivers. This could be all receivers or a suitable set of neighbours of $\mathbf{k}^{(*)}$. We estimate the probability distribution of $t^{(*)}$ to be normal with mean and variance

$$\begin{aligned} \bar{m} &= \underset{\mathbf{k}^{(i)} \in \mathcal{S}}{\text{mean}} \{t^{(i)}\}, \\ \bar{\sigma}^2 &= \underset{\mathbf{k}^{(i)} \in \mathcal{S}}{\text{var}} \{t^{(i)}\}. \end{aligned}$$

With this estimate, the expected change when considering $\mathbf{k}^{(*)}$ in the logarithm of the marginal likelihood is

$$\begin{aligned} E[\Delta\mathcal{L}] &= \int_{-\infty}^{\infty} \left[\ln \frac{1}{\sqrt{2\pi}\sigma_*} - \frac{(m_* - t^{(*)})^2}{2\sigma_*^2} \right] * \\ &\quad \frac{1}{\sqrt{2\pi}\bar{\sigma}} \exp\left(-\frac{(\bar{m} - t^{(*)})^2}{2\bar{\sigma}^2} \right) dt^* \\ &= \ln \frac{1}{\sqrt{2\pi}\sigma_*} - \frac{\bar{\sigma}^2 + (\bar{m} - m_*)^2}{2\sigma_*^2}. \end{aligned}$$

The second term is the important one. If the predictive probability distribution does not match well with the estimated probability distribution from the data, then the expected change in the logarithm of the marginal likelihood is negative. This expected change creates an uncertainty map with the largest negative values being the most uncertain regions. We will investigate the effectiveness of this method using seismic signals and comparing it with all the possibilities of the RVM. Next, we give a review of how the BPFA can create uncertainty maps.

7.2 Beta Process Factor Analysis and Gibbs samples

In section 5.2, we described the patch processing procedure followed in order to obtain more training data and help the inference stage of the Beta Process Factor Analysis (BPFA). A section is split into smaller overlapping patches, $\mathbf{x}^{(i)}$, which are used in a sequential manner. We chose a patch size of 8×8 in a grid of 128×128 receivers and shifted the extraction as described for all receiver locations in a given patch resulting in 64 such rounds. At each round, the extracted patches are used to perform Gibbs sampling over the unknown variables with more patches added sequentially until all rounds are completed. At every Gibbs round and every iteration, each variable is drawn from its distribution and used to calculate the receivers' value for all patches. Thus, each value is inferred various times since it is contained in numerous patches (at most 64). The mean (final prediction) of each receiver's value is obtained by averaging over all its estimated values. The uncertainty of the prediction at a receiver's location is obtained by calculating the variance of all its estimated values. To obtain the uncertainty map, we calculate the variance of all the values at a particular receiver's location for all locations.

Before moving on to the illustrations, we will discuss how the BPFA's uncertainty maps are affected by the speed up. In section 5.9, we proposed a speed up of the BPFA inference by using insight from our proposed Gibbs analysis. We will now see how this affects uncertainty. In order to evaluate this we need a metric for evaluation. The Spearman's correlation coefficient defined in equation 6.1 is able to capture the relationship between two variables. In the case of uncertainty, we would like to check the correlation between the variables that we use to create uncertainty maps (BPFA's variance) and their respective reconstruction errors. Thus, to evaluate the effect, we used twenty sections of time slices with only 50% of receivers, reconstructed them and then calculated the Spearman's correlation coefficient on the uncertainty and the respective reconstruction error. We tracked how it changes and plotted the mean Spearman's correlation coefficient against the computational time for the six different initialisations used in section 5.9.

The plot is given in Figure 7.1. We can see that at the beginning of the inference, the Spearman's correlation coefficient and consequently the uncertainty maps produced are poor. But after a few iterations, the Spearman's correlation coefficient increases dramatically and then it is not significantly affected over time. We can see that random initialisation performs the worse with a big difference with the rest of the initialisations. The others are similar with the SVD slightly worse. Nevertheless, we will use the SVD since it obtains the best reconstruction accuracy as seen in Figure 5.20. We will also

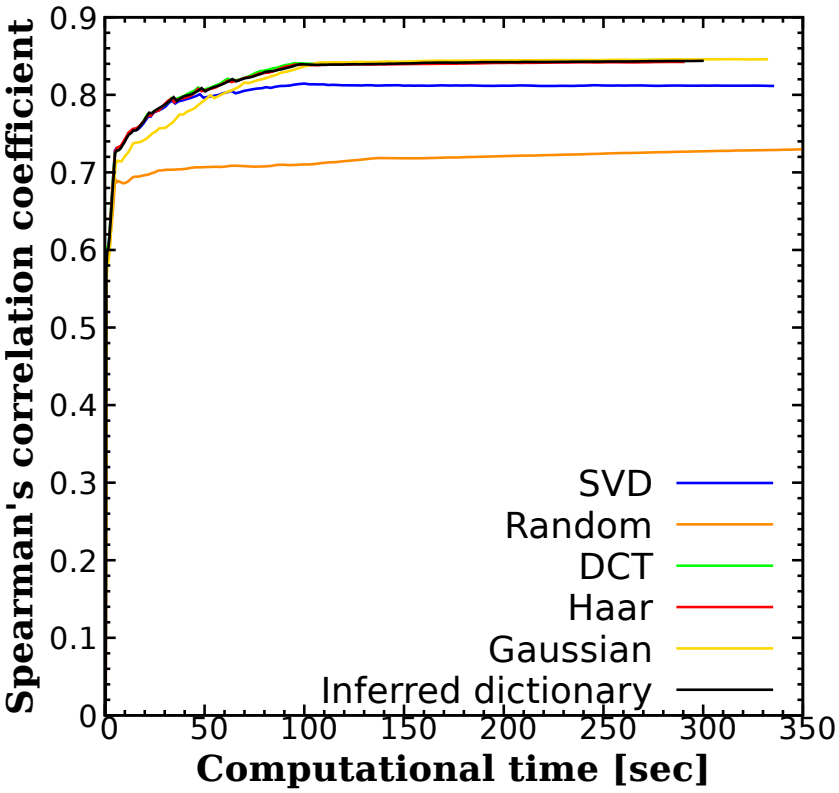


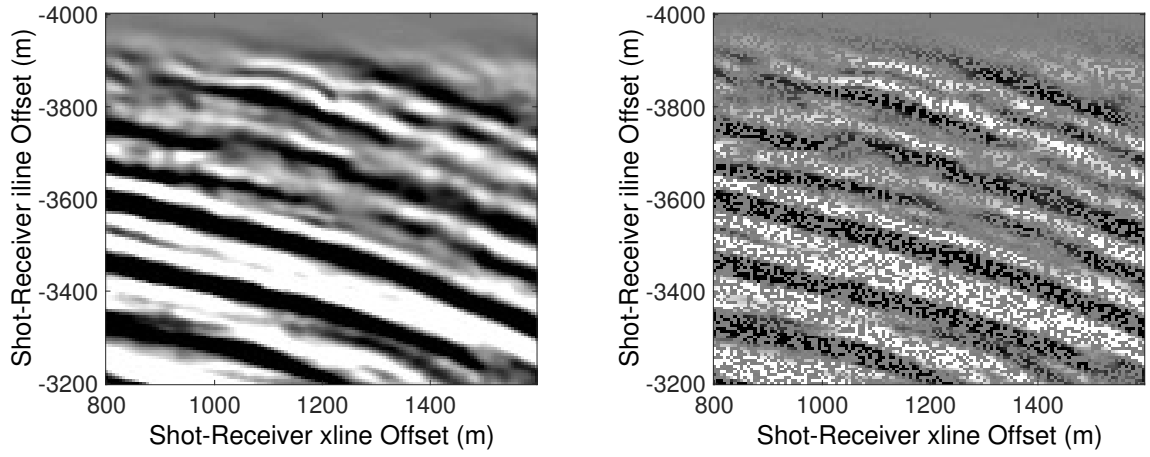
Fig. 7.1 Plot of the mean Spearman's correlation coefficient against computational time for various initialisation of the dictionary to be learned.

use all Gibbs rounds with the last round having 50 iterations saving in computation as discussed in section 5.9.

7.3 Uncertainty maps for seismic data

In order to create uncertainty maps, we used the 3D synthetic data set called SEAM-II described in chapter 2. One way to illustrate the effectiveness of the methods is to visualise the uncertainty maps produced along with the respective reconstruction error. Figure 7.2(a) shows the original section of a time slice. Figure 7.2(b) shows the signal using 50% of receivers. The receivers were muted randomly by going through the signal and drawing a random number between 1 and 100. If that number was below 50, it was kept otherwise not considered.

As discussed, the RVM uses a predefined dictionary of basis functions and we chose the Discrete Cosine Transform (DCT). Figure 7.3(a) shows the respective RVM reconstruction and Figure 7.3(b) the respective reconstruction error. Figure 7.3(c) shows the predictive variance of the RVM. We can see that the predictive variance captures some regions of



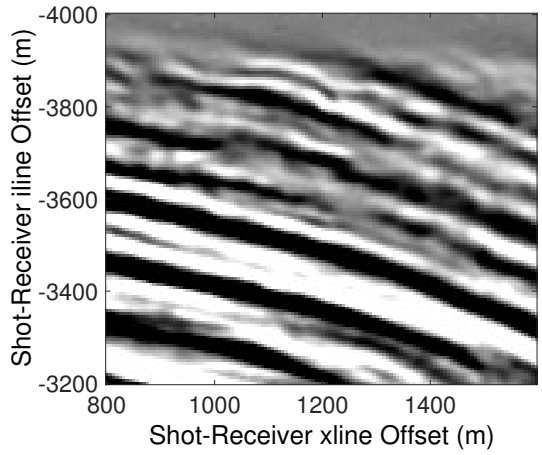
(a) Original signal

(b) Using 50% of receivers

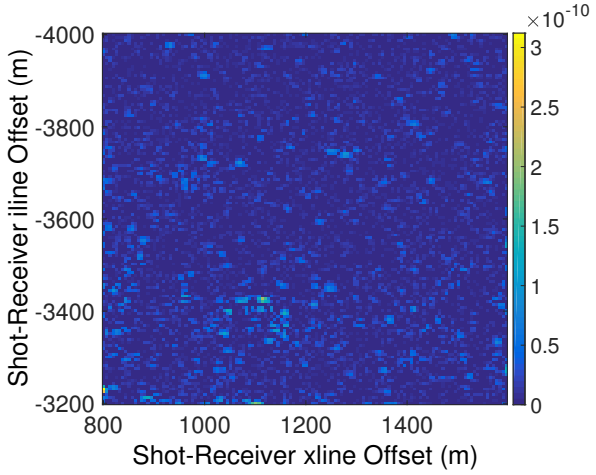
Fig. 7.2 Original used for uncertainty map (a) and using only 50% of receivers (b).

reconstruction error but in general it is spread around the domain. Figure 7.3(d) shows the expected change in the model shifted so that the higher the values the more uncertain we are. This is achieved by negating all the values and then adding the minimum value to each location. This shifts everything to the range of zero to positive values resulting in large values for uncertain areas. As it can be seen, it captures the variance of the signal with resemblance to the original. Nevertheless, it does not show any similarities with the reconstruction error produced by the RVM. Figure 7.3(e) shows the predictive variance with augmentation using Gaussian basis functions at test points. The uncertainty map is similar to the original predictive variance since the basis functions used (DCT) are not prone to the degenerate case. The same signal is used to produce an uncertainty map using BPFA. Figure 7.4(a) shows the BPFA's reconstruction. The respective reconstruction error can be seen in Figure 7.4(b). The uncertainty map produced by BPFA can be seen in Figure 7.4(c). The learned bases dictionary is illustrated in Figure 7.4(d) which captures the direction of the largest variations in the signal.

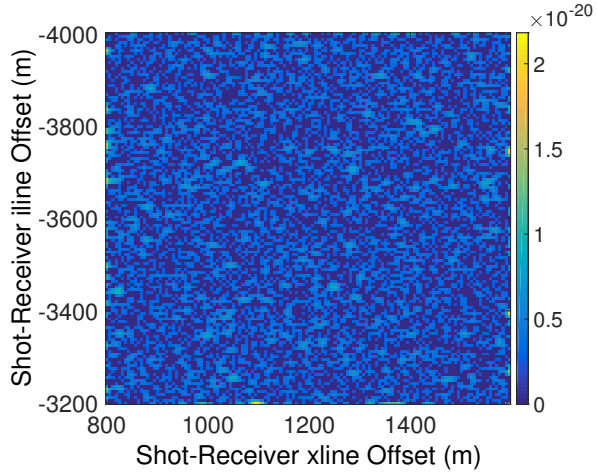
The uncertainty map produced by BPFA shows good correlation with the error and is more informative than the rest. We will see later how we can quantify this. In addition, we will illustrate scatter plots to visualise the relationship between the various uncertainty maps and their respective reconstruction errors. Note that we will not follow the comparison with the augmentation and the expected change in model likelihood since, for a fair comparison, it requires extensive parameter tuning such as the choice of the new basis function and the parameter that it depends (i.e. for Gaussian basis function, λ_d) and is not the purpose of this thesis.



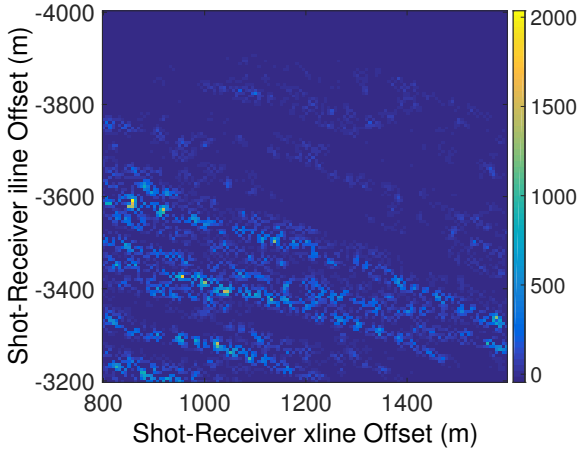
(a) RVM reconstruction



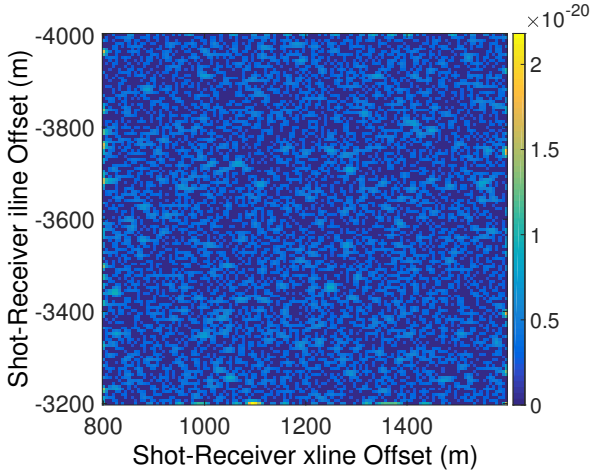
(b) RVM's error



(c) Predictive variance

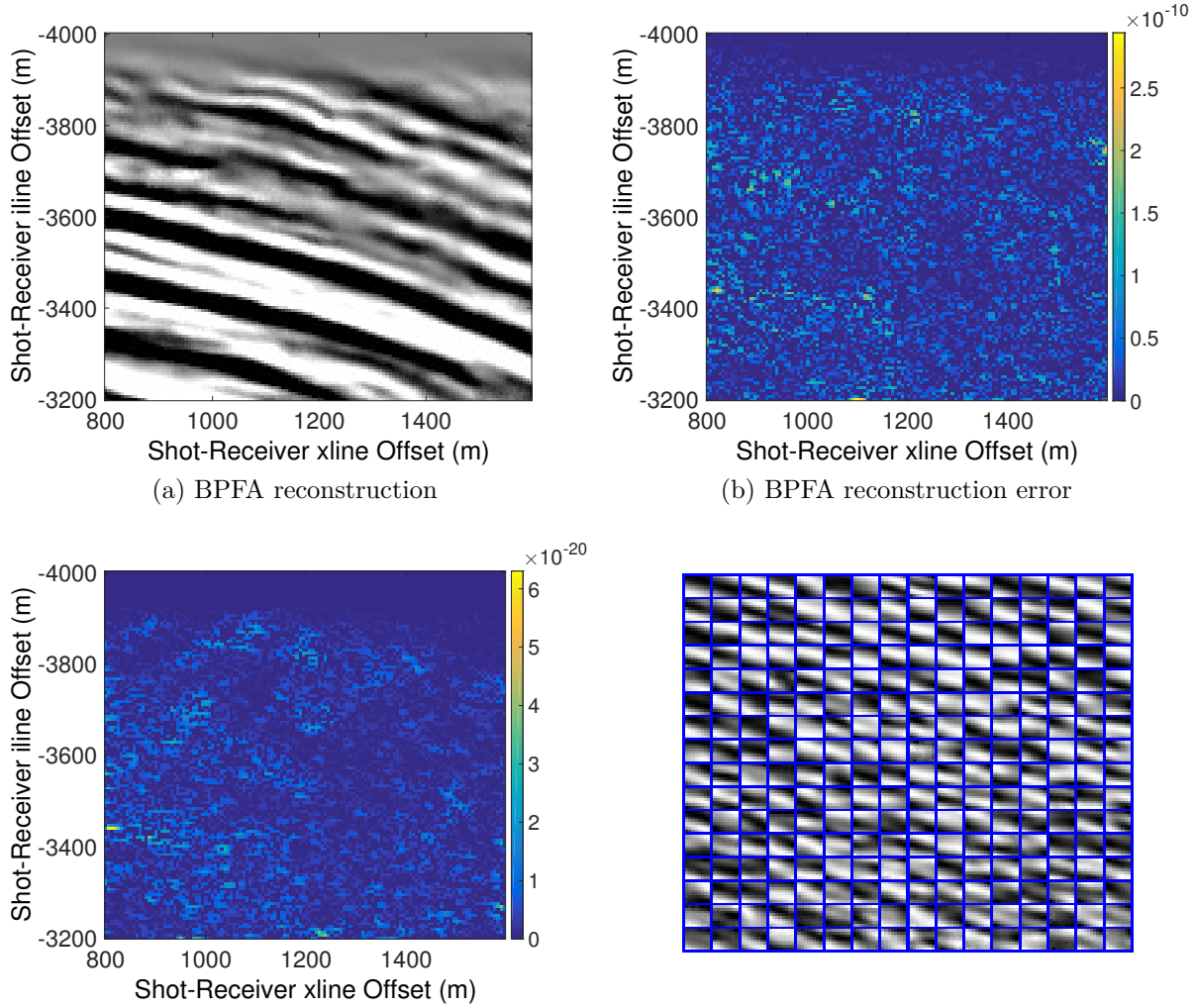


(d) Shifted expected change



(e) RVM with augmentation

Fig. 7.3 The RVM with DCT (a), the reconstruction error (b), the uncertainty map using the RVM's original predictive variance and DCT (c) and with augmentation (e). The expected change in the model in (d).



(a) BPFA reconstruction

(b) BPFA reconstruction error

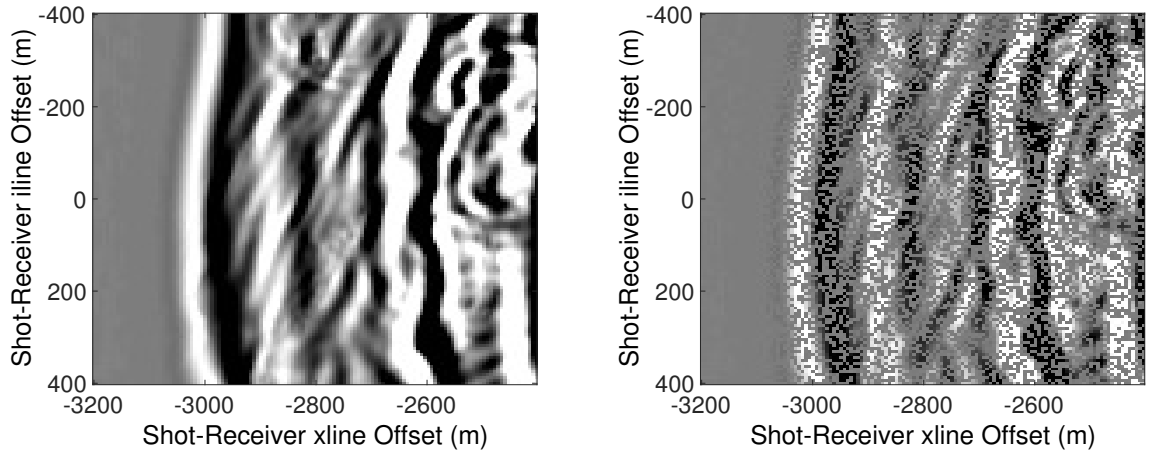
(c) BPFA's variance

(d) BPFA's bases

Fig. 7.4 The BPFA reconstruction (a), the error in (b), the BPFA uncertainty map in (c) and the respective learned bases dictionary in (d).

7.4 Uncertainty quantification using the Spearman's correlation coefficient

Following the time slice processing approach in chapter 4 and 5, we process sections from consecutive time slices. In order to obtain uncertainty maps using time slice processing, we randomly removed receivers from 10000 sections (5000 for far source receiver lines and 5000 for close to source receiver lines) of time slices of 128×128 patch size and then reconstructed them. That is, 20 sections per time slice for a total of 500 time samples.



(a) Original signal

(b) Using 50% of receivers

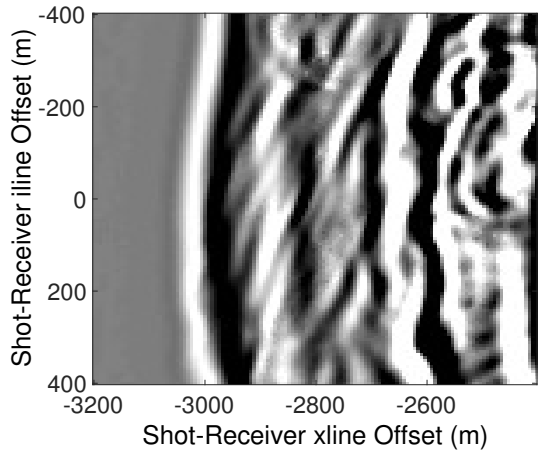
Fig. 7.5 Original (a) and 50% of receivers (b).

This was done for both the Relevance Vector Machine (RVM) and the Beta Process Factor Analysis (BPFA) for different percentages of receivers.

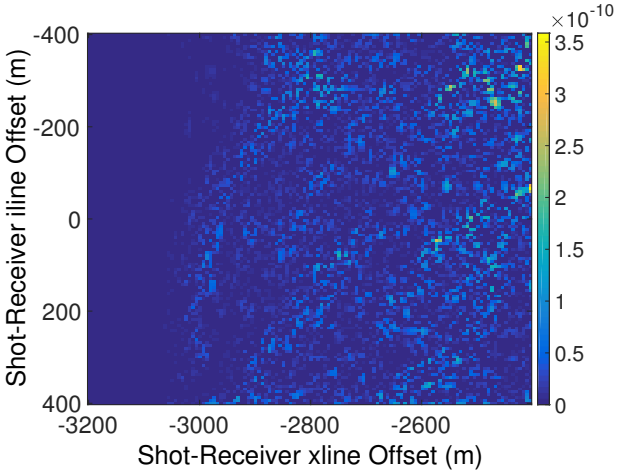
From the visualisations in the previous section, it can be seen that BPFA's variance creates more informative uncertainty maps. In order to evaluate over various signals and scenarios, we need a quantitative metric. With this, we can compare the algorithms and also get a better understanding of the methods. To do this, we propose to check how much the reconstruction error correlates with the uncertainty map in a corresponding receiver location. That is, ideally, the larger the reconstruction error in a data point the larger the uncertainty and vice versa.

Scatter plots for uncertainty

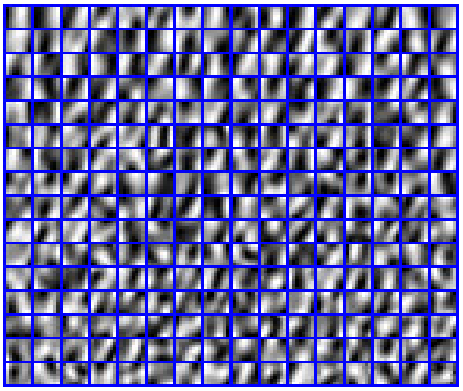
In order to understand the relationship between reconstruction error and uncertainty, we first illustrate a direct scatter plot between the BPFA's variance and the respective reconstruction error. To do this, we use a signal from closer to the source as seen in Figure 7.5. Figure 7.5(a) shows the original signal with only 50% of receivers shown in Figure 7.5(b). The BPFA reconstruction is given in Figure 7.6(a) and the respective reconstruction error in Figure 7.6(b). The learned dictionary of bases is given in Figure 7.6(c) showing that it captures the orientation of the largest changes in the seismic signal. The uncertainty map produced by the BPFA is given in Figure 7.6(d) which correlates well (visually) with the reconstruction error.



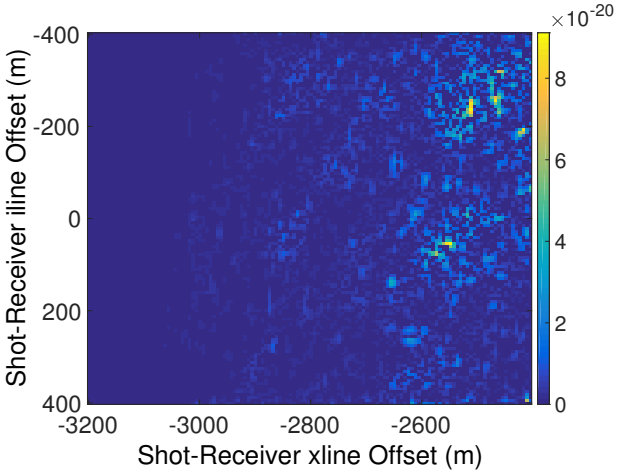
(a) BPFA's reconstruction



(b) BPFA's reconstruction error



(c) BPFA's bases

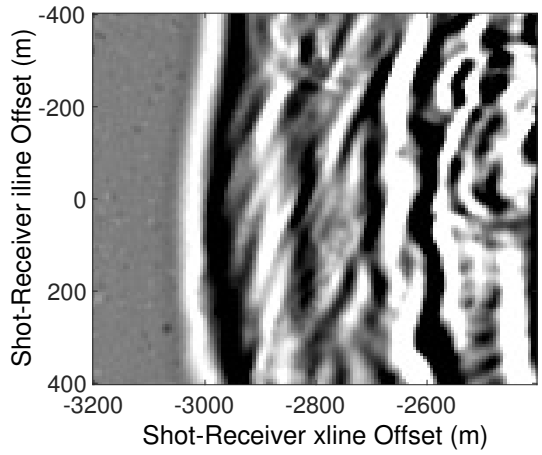


(d) BPFA's variance

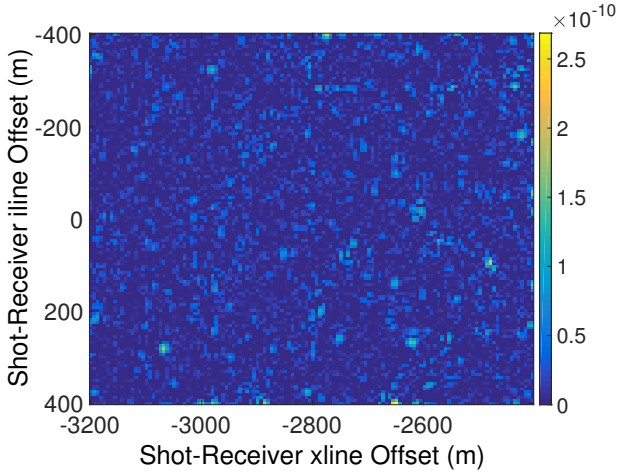
Fig. 7.6 BPFA's reconstruction (a), the reconstruction error (b), the learned bases (c) and the BPFA's variance (d) using the seismic signal of Figure 7.5.

For the RVM, the reconstruction is given in Figure 7.7(a), the reconstruction error in Figure 7.7(b) and the RVM's predictive variance is shown in Figure 7.7(c). We can see that again, visually, the BPFA is better than the RVM.

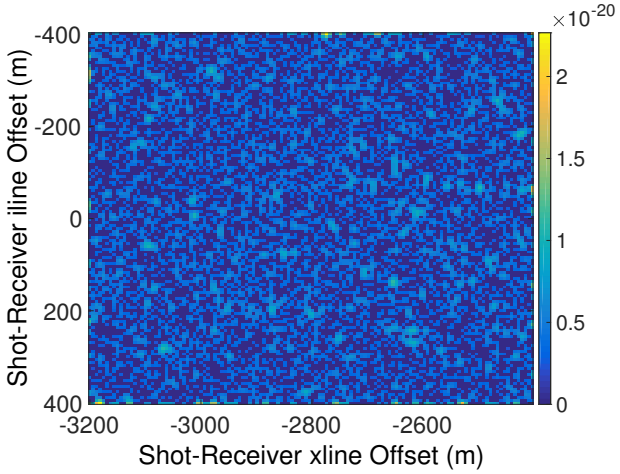
To get an understanding of the correlation of the uncertainty and the error for each algorithm, we produce scatter plots of these variables. First, in Figure 7.8, the BPFA's variance against the BPFA's reconstruction error is plotted for the signal in Figure 7.6. We can see that a lot of data points are concentrated near the origin and it is not clear how the two variables are correlated. Therefore, we transform them to their ranked version depending on their magnitude (equal values are ranked equally).



(a) RVM's reconstruction



(b) RVM's reconstruction error



(c) RVM's predictive variance

Fig. 7.7 RVM's reconstruction (a) and its reconstruction error (b) and the RVM predictive variance (c) using the seismic signal of Figure 7.5.

The ranked data points are now plotted for both methods of interest, namely the BPFA's variance and the RVM's predictive variance. Figure 7.9 shows the same data points as in Figure 7.8 but now ranked according to their value. This allows the data points to spread out instead of being near the origin. The respective scatter plot for the the RVM's predictive variance is given in Figure 7.10. For each scatter plot, we provide the corresponding Spearman's correlation coefficient. For the BPFA, this coefficient is equal to 0.7064 illustrating that there is a strong positive correlation between the variables. The RVM's coefficient is equal to 0.3514 which shows a weaker correlation. Comparing the two scatter plots visually, we can see that the BPFA's variance correlates much better with the reconstruction error as inferred quantitatively.

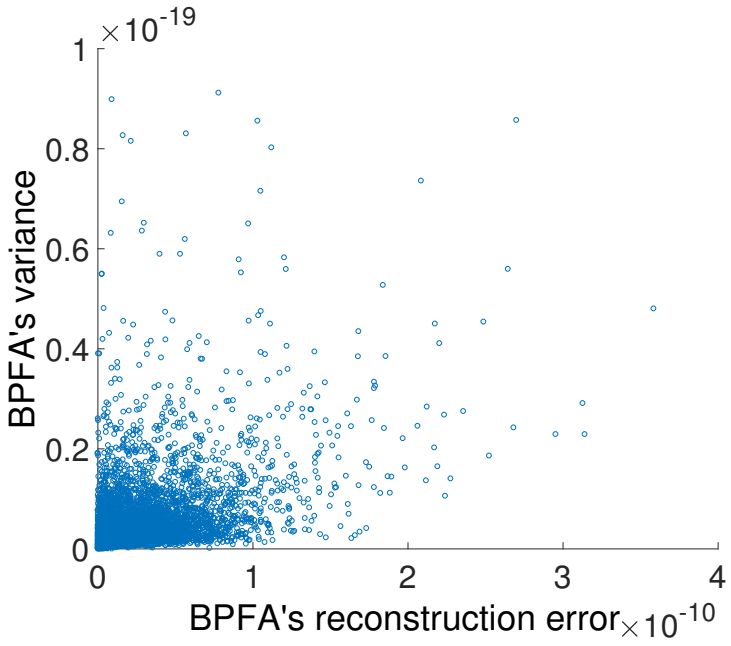


Fig. 7.8 Direct scatter plot between the BPFA's variance and the respective reconstruction error.

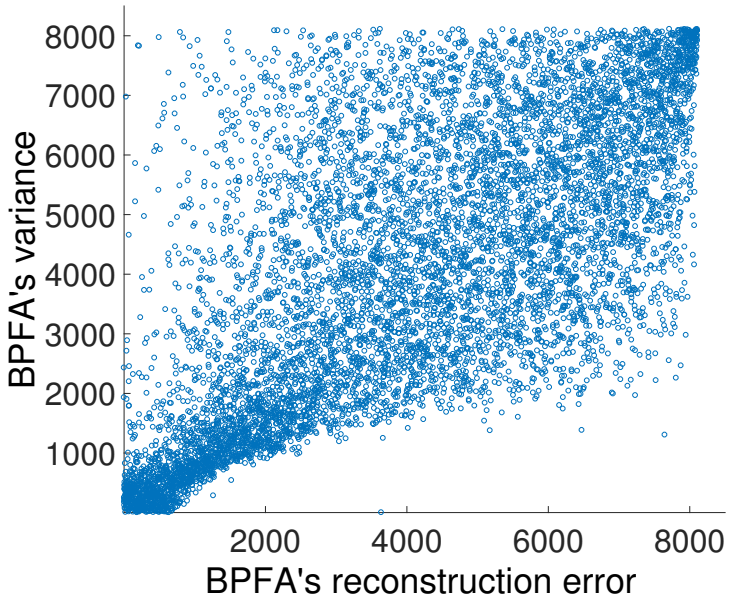


Fig. 7.9 Ranked scatter plot for BPFA's variance against the respective reconstruction error with the Spearman's correlation coefficient as defined in equation 6.1 equal to 0.7064.

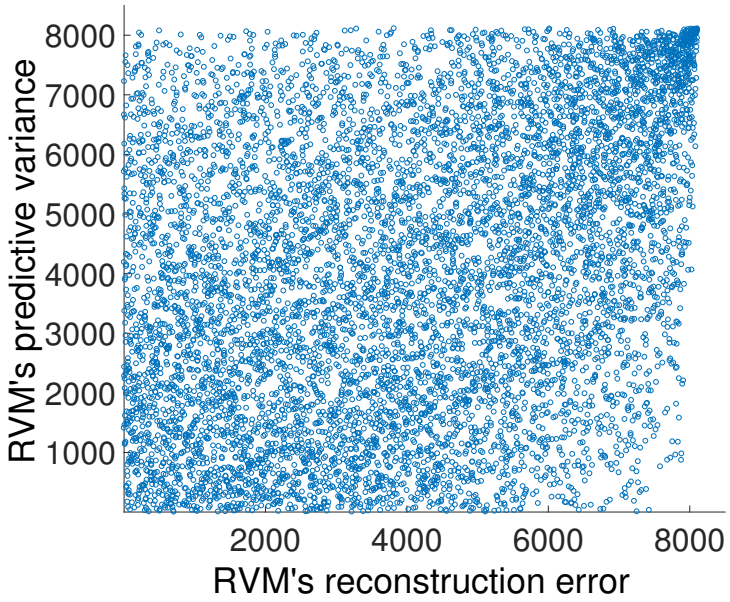


Fig. 7.10 Ranked scatter plot for RVM's predictive variance against the respective reconstruction error with the Spearman's correlation coefficient as defined in equation 6.1 equal to 0.3514.

Analysis using the Spearman's correlation coefficient

To evaluate the performance over numerous sections of time slices, we will use the Spearman's correlation coefficient as defined in equation 6.1. Note that each variable is ranked as we have seen in Figures 7.9 and 7.10. The Spearman's correlation coefficient ranges from -1 to $+1$. Positive correlation means that as one grows so does the other. On the other hand, if the other decreases in value, it means that it has negative correlation. In this chapter, we would like to check the correlation between the variables that we use to create uncertainty maps (BPFA's variance and RVM's predictive variance) and their respective reconstruction errors.

7.5 Comparisons for uncertainty quantification

As discussed, we used 5000 sections of time slices far from the source and 5000 closer to the source as seen in Figure 2.6. Using three percentages $(30\%, 50\%, 70\%)$ of receivers, we use the RVM and the BPFA to reconstruct the signals and then create uncertainty maps. We split the evaluation for far from source signals and close to source signals and

Table 7.1 Mean uncertainty quantification for 2000 sections (1-200 time samples) of far source signals.

Spearman's correlation coefficient			
Percentage used	30%	50%	70%
RVM's predictive variance	0.0027	0.0021	0.0012
BPFA's variance	**0.4092**	**0.4056**	**0.3724**

Table 7.2 Mean uncertainty quantification for 3000 sections (201-500 time samples) of far source signals.

Spearman's correlation coefficient			
Percentage used	30%	50%	70%
RVM's predictive variance	0.2779	0.2962	0.1967
BPFA's variance	**0.5254**	**0.5221**	**0.5337**

also for different time samples using the insight from section 6.4 and Figures 6.28 - 6.30 from the variance analysis.

Table 7.1 shows the mean Spearman's correlation coefficient, calculated for both uncertainty methods along with three different percentages for the first 2000 sections of time slices. For these results, we used the sections of time slices when $t <= 200$, there are many sections that include almost zero signal, the receivers have not yet seen the signal. It is clear that the BPFA's variance is positively correlated with the reconstruction error with a higher correlation coefficient than the RVM. In addition, it is only slightly affected by the percentage of the available receivers used provided that there are enough training data to learn a dictionary of bases. On the other hand, the RVM's predictive variance correlation is very close to zero. Table 7.2 shows the average Spearman's correlation coefficient for the last 3000 sections of time slices when $t > 201$. In this region, there are sections with larger variance (refer to Figure 6.30). The correlation is higher for both algorithms in general due to the presence of signals with higher variance. The BPFA's variance is better and not affected a lot by the percentage as opposed to the RVM's predictive variance.

Moving on to closer to source sections of time slices, Table 7.3 shows the average Spearman's correlation coefficient for 1000 sections when $t < 101$. This region is characterised with very high and low variances as seen in Figure 6.28. We can see that the BPFA's variance is still better. Table 7.4 shows the average correlation for

Table 7.3 Mean uncertainty quantification for 1000 sections (1-100 time samples) of close to source signals.

Spearman's correlation coefficient			
Percentage used	30%	50%	70%
RVM's predictive variance	0.0718	0.1477	0.1174
BPFA's variance	**0.5804**	**0.5169**	**0.5083**

Table 7.4 Mean uncertainty quantification for 4000 sections (101-500 time samples) of close to source signals.

Spearman's correlation coefficient			
Percentage used	30%	50%	70%
RVM's predictive variance	0.2904	0.3003	0.2275
BPFA's variance	**0.3804**	**0.3632**	**0.3685**

the rest 4000 sections when $t > 100$. The BPFA's variance is again better with the RVM's predictive variance obtaining improved results. From the results of quantifying uncertainty both for far from the source and closer to the source, we can see that the BPFA's performance is better in all cases. However, its performance varies depending on the region of the signal that it operates.

7.6 Variance analysis for uncertainty quantification

The variance of the Gibbs samples of the BPFA inference provide the most correlated results with the respective reconstruction error as opposed to the RVM. Nevertheless, we also want to identify how the correlation behaves as the variance of the available data changes. Thus, we provide a variance analysis using the Spearman's correlation coefficient, similar to the one provided in section 6.4 for the reconstruction accuracy.

We have obtained 30000 reconstructions of sections of time slices (5000 far and 5000 close to the source for three different percentages) for both algorithms. For brevity, we include only analysis for 50% of receivers used. For each method, a scatter plot is provided for the Spearman's correlation coefficient against the variance of the available data used per section. Figure 7.11 shows the scatter plot of the Spearman's correlation coefficient for the BPFA against available variance of data. We can see that the scatter plot is wide in general. This means that the uncertainty maps do not heavily depend

on the variance. The Spearman's correlation coefficient of the overall scatter plot (i.e. showing how much the Spearman's correlation coefficient depends on the variance of the available data) is equal to -0.0179 which is negligible. Nevertheless, in a region of the scatter plot between 1 to 1500 for the rank of the variance, the correlation coefficient is low when the variance of the available data is low. This is expected since when there are not enough variations, BPFA does not learn a useful dictionary of bases but learns random bases. This produces poor reconstruction and thus non-informative uncertainty.

We repeat the same scatter plot but this time for the RVM's predictive variance against the variance of the available data in Figure 7.12. This scatter plot has significant differences to the respective scatter plot of the BPFA's variance. There are approximately 3000 sections of time slices that produce the same value for the Spearman's correlation coefficient. This value occurs when the denominator of equation 6.1 is zero. This does not produce a number and we thus set it to zero manually. It occurs when all the receivers in the reconstruction have the same value and thus their mean is also the same producing an exact zero in the denominator. This happens when the variance is very small and the model is very simple giving the same value for all receivers. As the variance of the available data increases, the Spearman's correlation coefficient increases with the overall correlation coefficient for the two variables in the scatter plot equal to 0.8360. This shows that the two variables are highly correlated.

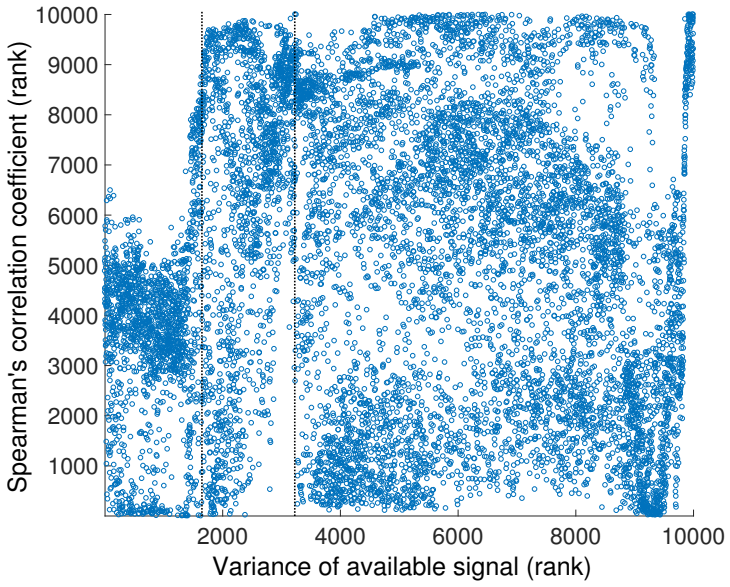


Fig. 7.11 Spearman's coefficient for BPFA against variance of available data. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

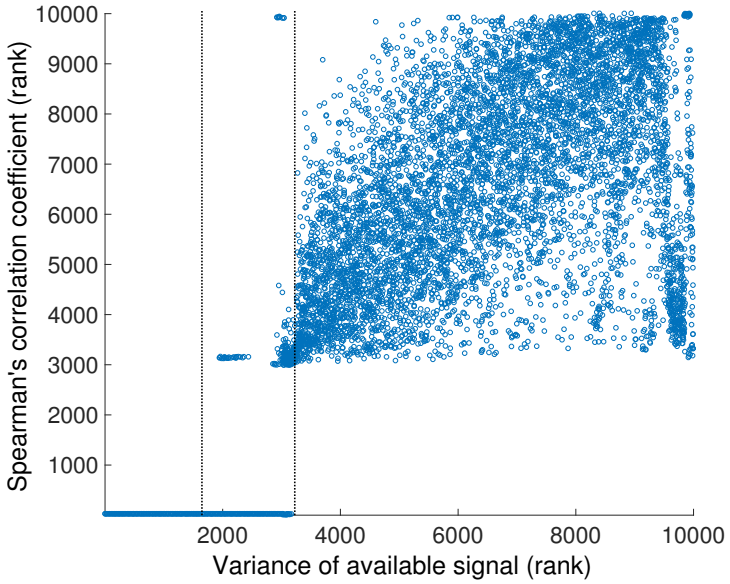
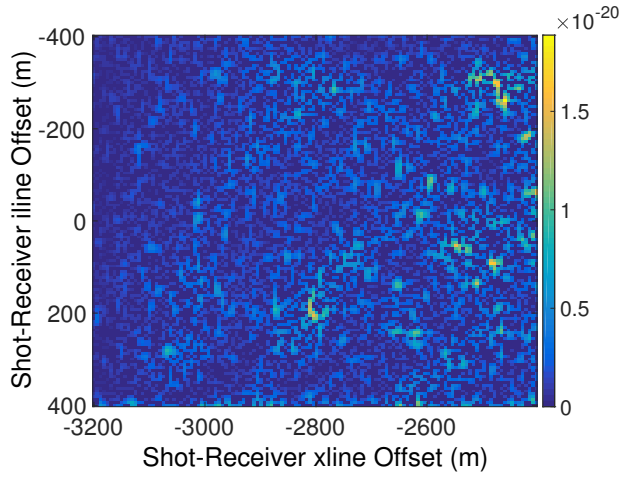


Fig. 7.12 Spearman's coefficient for the RVM's predictive variance against variance of available data. Region on the left of the first line includes sections of very low variance (no reflected signal). Region between the two lines includes sections with no or almost no reflected signal. The region on the right of the second line includes sections with the most reflected signal.

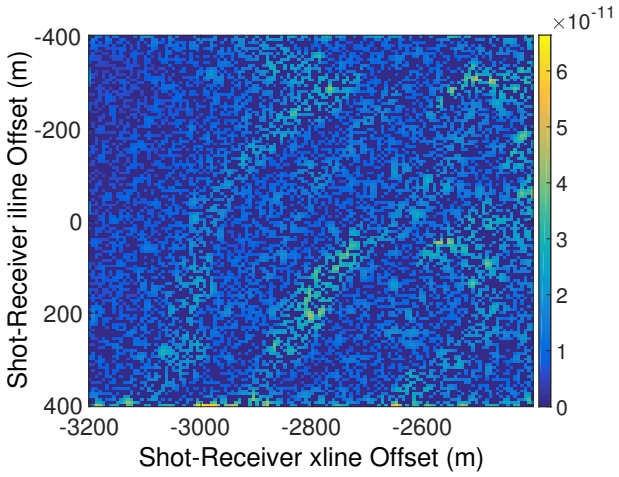
7.7 Stacking of uncertainty maps

From the previous section, we have seen that the correlation of the uncertainty maps with the respective reconstruction errors varies with different variances of available data. Viewing the uncertainty map at one time sample is useful, nevertheless, it does not provide the complete uncertainty since each receiver has 500 time steps of varying correlation levels. In order to get a complete understanding of the uncertainty for a receiver, it is useful to take into account all time samples associated with it. One option would be to sort the uncertainty into the x-t domain as it was done for reconstructions. Nevertheless, this will not provide a quantitative metric for the complete uncertainty. Therefore, we decided to stack all 500 uncertainty maps for each receiver together and take the average value per receiver location.

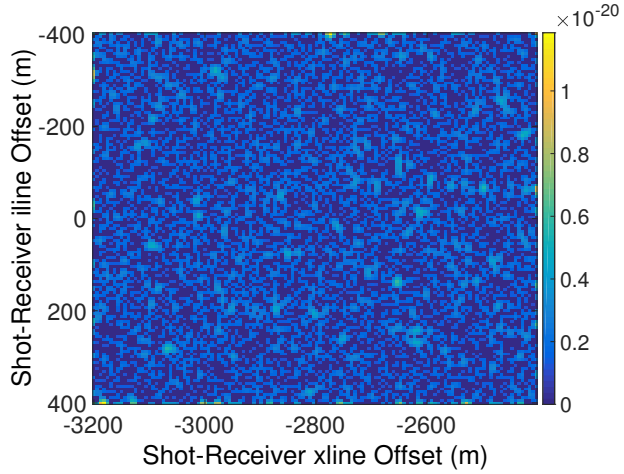
Figure 7.13 shows various stacked uncertainty maps for a section of receivers from a time slice in the same location as the section in Figure 7.6 but using all 500 time samples. We can see the BPFA's stacked variance in Figure 7.13(a) and the respective stacked reconstruction error in Figure 7.13(b). The uncertainty map picks up regions of large reconstruction error with a Spearman's correlation coefficient equal to 0.8306. This is higher than the individual uncertainty maps since the averaging of uncertainties helps.



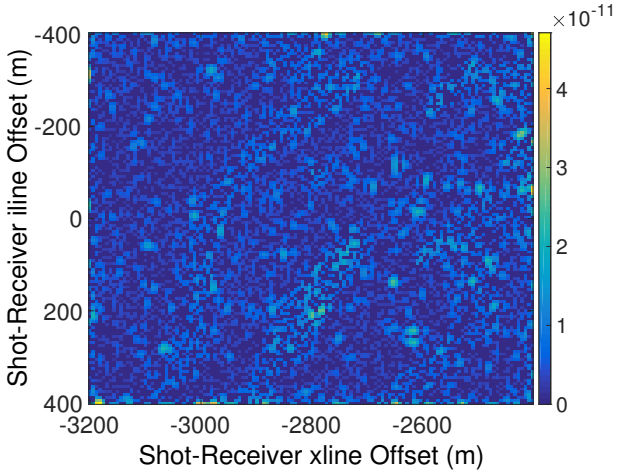
(a) BPFA's stacked variance (Spear = **0.8306**)



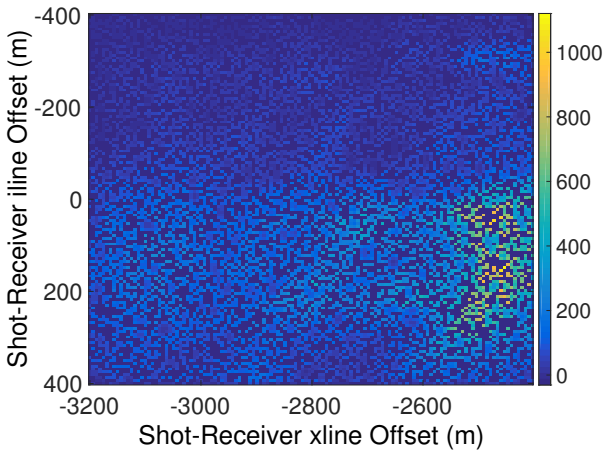
(b) BPFA's stacked reconstruction error



(c) RVM's stacked predictive variance (Spear = 0.7526)



(d) RVM's stacted reconstruction error



(e) RVM's stacked expected change (Spear = −0.1460)

Fig. 7.13 Average uncertainty for different methods using all 500 uncertainty maps produced per time samples for each receiver.

Table 7.5 Mean uncertainty quantification of 20 sections stacked with 500 uncertainty maps (1-500 time samples) per percentage

Spearman's correlation coefficient			
Percentage used	30%	50%	70%
RVM's predictive variance	0.8503	0.7164	0.6185
BPFA's variance	**0.8533**	**0.9082**	**0.9168**

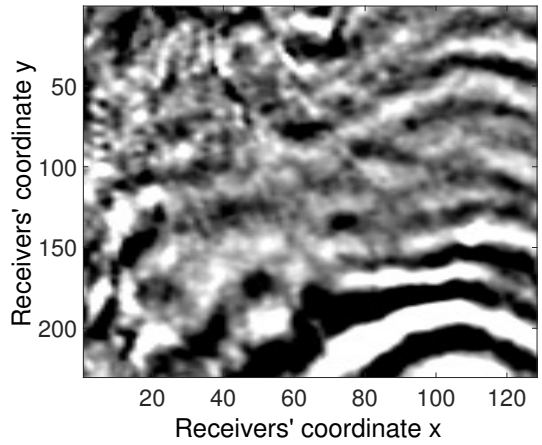
The RVM's stacked predictive variance is included in Figure 7.13(c) with the respective stacked reconstruction error in Figure 7.13(d). We can see that the RVM's uncertainty map is also significantly improved providing higher correlation with the error with a Spearman's correlation coefficient equal to 0.7526. The averaging helps amplify uncertainties which are correctly calculated and diminishes the randomness in uncertainties. For completeness, we include the stacked expected change of the RVM in Figure 7.13(e). It shows that the stacking does not help in this case since each individual uncertainty map is badly produced. Stacking them does not produce a better result with a negative correlation coefficient equal to -0.1460. Better tuning could improve its result but as mentioned before, this is not the purpose of this thesis.

In order to get a better understanding of the performance of the BPFA's stacked variance and the RVM's stacked predictive variance, we repeat the experiment for all 20 sections of time slices (10 far from source and 10 close to source) averaging their 500 uncertainty maps. Table 7.5 shows the average Spearman's correlation coefficient for these 20 stacked sections. The BPFA's variance provides very high correlation with the stacked reconstruction error showing the improvements obtained with averaging. The same is true for the RVM's predictive variance that also provides very high correlation. This correlation increases as the percentage of receivers decreases. This is due to the fact that there is more reconstruction error with less receivers and at the same time the uncertainty increases correctly. Overall, the BPFA's variance still produces better uncertainty maps compared to the RVM's predictive variance.

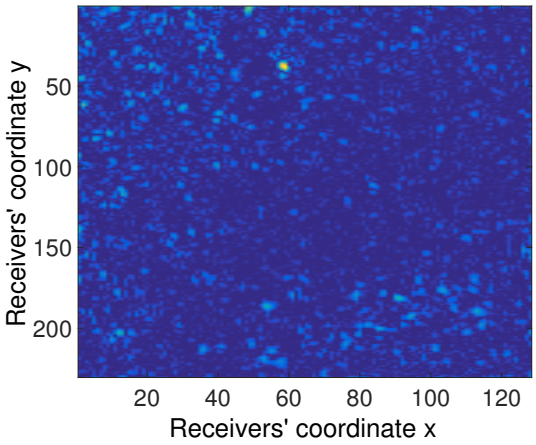
7.8 Uncertainty maps for field data

We will now illustrate the BPFA's performance on field data. We will use the Parihaka data set which is a 3D seismic image provided for use by New Zealand Petroleum and Minerals (NZPM) (SEG, 2018b). A section from a time slice is processed, using only 50% of receivers. Figure 5.30(a) used in chapter 5 shows an original section from the Parihaka

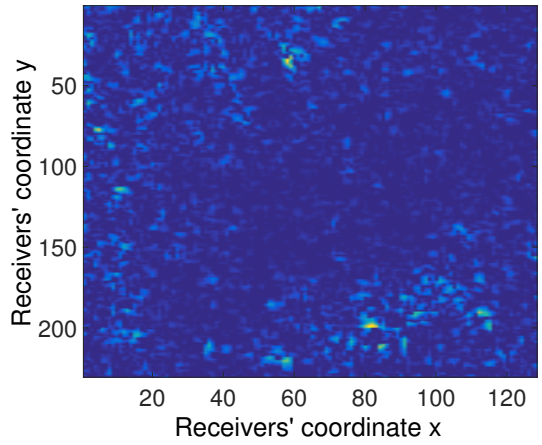
data set. Figure 5.30(b) shows the same signal with 50% of the receivers used and Figure 7.14(a) shows the BPFA's reconstruction. Figure 7.14(b) shows the reconstruction error and Figure 7.14(c) shows the variance of BPFA's samples with a Spearman's correlation coefficient equal to 0.4755. Figure 7.14(d) shows the learned bases from the available data. We can see that BPFA learns a dictionary of bases that captures the important features in the data, reconstructs the signal well and the reconstruction error and the uncertainty map are positively correlated. This illustrates its effectiveness to more complex signals found in field data.



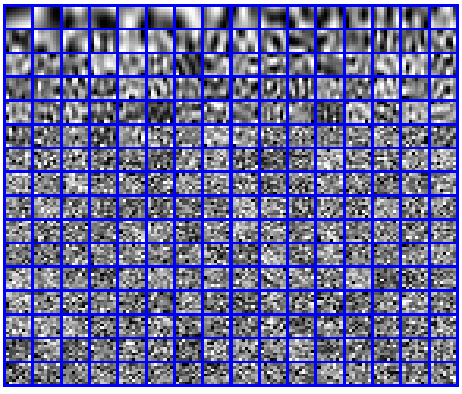
(a) BPFA's reconstruction



(b) BPFA's reconstruction error



(c) BPFA's variance



(d) BPFA's bases

Fig. 7.14 The BPFA's reconstruction (a), the BPFA's error (b), the BPFA's variance with Spear = 0.4755 (c) and the learned bases (d) for the section in Figure 5.30.

Chapter 8

Discussion and conclusions

Sampling the seismic wave field during seismic surveys and at the same time obtaining the true underlying signal has many challenges. This becomes even more difficult when there are physical, environmental or financial limitations in the placement of receivers and/or artificial sources. Nevertheless, if seismic surveys are undertaken more efficiently, the potential gains are enormous for the environment, the health and safety conditions of the survey and its financial cost. In this thesis, we proposed to reduce the number of receivers and then process the remaining seismic data to reconstruct the missing values by working on the common-shot domain.

Seismic Compressive Sensing (CS) and interpolation is the field that tackles this problem. Current state-of-the-art algorithms are able to reconstruct under-sampled seismic signals but there are many limitations and challenges yet to be solved. One reason for this is that algorithms are ad hoc, created for solving a specific task such as filling gaps in between receivers. Instead of using such algorithms, we proposed to create probabilistic data-driven models that can be used for various purposes such as: interpolation, denoising, feature learning and uncertainty quantification. This is possible because we used the available data during training to adjust the data-driven models. Using the Relevance Vector Machine (RVM) and the Beta Process Factor Analysis (BPFA), we addressed the several challenges in the seismic CS field.

Using fewer receivers and at the same time ensuring that the reconstruction accuracy is of sufficient quality and resolution is a challenge. Reducing the number of receivers with regular under-sampling can cause aliasing which creates problems for the rest of the seismic processing work flow (Naghizadeh and Sacchi, 2010). Instead, in this thesis, we used irregular under-sampling to reduce the number of receivers randomly. This irregularity does not introduce aliasing but rather incoherent noise in the Frequency-

Wavenumber (FK) domain. Missing receivers in the time slice and in the x-t domain produce different gaps in the data. For the former, a missing receiver is a data point and for the latter, a missing receiver is an entire line of data points missing. Both the RVM and the BPFA perform better when operating on time slices due to the fact that the training data are balanced without large gaps missing. We thus followed a time slice processing approach where sections from time slices were reconstructed over time in order to reconstruct three-dimensional signals with success. Working directly in three dimensions was deemed impractical with the current implementations.

The first probabilistic data-driven model used was the RVM which is suitable for seismic CS since it is composed of a linear combination of basis functions with only a few non-zero coefficients. This results in a sparse model, an assumption necessary for CS to work. By using a prior normal distribution promoting sparsity (zero mean) and a normal likelihood function, the posterior distribution for the coefficients is also normal and obtained via an inference procedure. By using this posterior distribution, a predictive distribution is created and is used for prediction and uncertainty quantification.

Nevertheless, to train an RVM model requires extensive parameter tuning due to its numerous configurations. First, the noise standard deviation of the likelihood function was tuned by experimenting with eleven different values. The noise standard deviation of 10^{-11} was chosen for the experiments which is an order of magnitude smaller than the standard deviation of the available data. Then, a trade-off analysis of various configurations of the RVM was performed, varying the patch size that it operates, the basis functions and the number of RVMs used sequentially. From this, we found that the larger the patch size, the better the reconstruction accuracy but also the longer the computational time. In addition, using the Discrete Cosine Transform (DCT) with only one RVM gives the best reconstruction accuracy as opposed to using the Haar wavelets in a sequence of many RVMs. We thus decided to use the RVM with DCT on 128×128 patches to compare against POCS and SPGL1.

Before comparing the RVM with these algorithms, we investigated some of their configurations. From the POCS experiments, the performance with the best reconstruction accuracy was obtained when the patch sizes and the number of iterations were the largest. Larger patch sizes contain more signal structure and this allows the algorithm to use more information for reconstruction. Running for longer allows POCS to reconstruct more signal details, both high and low frequency components. However, from our experiments, the difference in reconstruction was not large enough to deem the extra computational time necessary. This could be the case when using more dimensions. During the SPGL1 experiments, the convergence criteria were fixed but we performed some preliminary

experiments on the residual tolerance and suggest that the residual should be orders of magnitude smaller than the l_2 norm of the available data. Another technical insight that might improve the SPGL1 is the formulation of the problem in equation 2.27. Collapsing the signal leads to the loss of location information and changing this could be useful in the reconstruction.

Using the selected configurations, we compared the RVM with DCT on 128×128 patches against SPGL1 with DCT and POCS both in reconstruction accuracy and in computational time on various percentages. The RVM obtained the best reconstruction accuracy but also took the longest to run. Smaller patch sizes can improve the computational time but the reconstruction accuracy is also degraded. In order to avoid degradation when operating in smaller patch sizes, BPFA helps by learning a dictionary of bases from the available measurements at the same time as interpolating the seismic data. By exploring all possible spaces, BPFA is able to learn a sparse representation that captures the signal variations of the seismic data and provides higher quality of reconstruction compared to other algorithms with predefined basis functions.

The BPFA model is also composed of a linear combination of bases with a few non-zero coefficients. In this case, the coefficients are modelled by two variables. One variable models whether the coefficient is non zero or zero with a Bernoulli prior distribution and the other variable models the value of the coefficient with a normal prior distribution. This way, the coefficients are truly sparse with exact zero elements as opposed to very small values in the RVM model. In addition, a prior normal distribution on the dictionary of bases is assumed. By obtaining conditional posterior distributions for these and other variables of the model, we use Gibbs sampling to infer their values.

Due to the fact that the learning of bases is done simultaneously with reconstruction, BPFA uses a distorted version of the seismic signals. However, dropping or adding noise in the training data is in fact recommended as regularisation (Beckouche and Ma, 2014; Vincent et al., 2008). The percentage of receivers dropped from the training data is important. In our experiments, we found that omitting more than 75% of the receivers does not allow sufficient training data for BPFA to learn a useful dictionary of bases. With more receivers available the reconstruction accuracy is higher, but, the bases learned are not necessarily the most informative. Varying the percentage of receivers used and evaluating the bases learned is an interesting future research question.

The BPFA operated on 8×8 patches since the computational cost for larger patch sizes is much higher and would be impractical to increase it. We compared this with the RVM, the SPGL1 and POCS on both 8×8 and 128×128 patches. For BPFA, extensive parameter tuning was not necessary due the adaptability achieved by learning bases.

The initialisation of the noise precision for BPFA was done based on the inverse variance of the available data in a similar fashion to the RVM. Wrong estimation of the noise precision leads to under-fitting in which case the algorithm would assume that early termination is justified since variations are explained by noise.

To evaluate the performance of BPFA, we compared the same algorithms as before on 8×8 and 128×128 patches. The BPFA obtained the highest reconstruction accuracy out of all algorithms on 8×8 patches. In addition, it is better than the SPGL1 with DCT and from POCS on 128×128 patches. The SEAM-II data set contains signals with varying structures and different variance, containing both high and low frequency characteristics. When there are insufficient receivers available (i.e. less than 60%), the fixed basis functions used by POCS and SPGL1 do not capture all these variations, especially the high frequencies. BPFA is able to adapt the bases and is able to capture the high frequencies (the details of the signals), resulting in higher quality of reconstruction. Nevertheless, it is worth mentioning that the dictionary of bases learned by BPFA is not optimum for the signal at hand. The optimisation problem solved is non-convex with only a local solution obtained. Different initialisation provides different bases and is thus very sensitive to the starting point as seen in the Gibbs analysis. However, the set of bases in practice yields a significant increase in reconstruction accuracy. We also investigated the performance of the BPFA on missing artificial rivers and missing blocks of receivers with success. The same is true when operating on field data where the signals are more complex by learning appropriate dictionary of bases.

The learned dictionary of bases by the BPFA can be used by other algorithms as well on 8×8 patches. We proposed two hybrid configurations, namely the SPGL1 with the BPFA bases and the RVM with the BPFA bases. We performed experiments on 10000 sections for three different percentages of receivers to test the algorithms on 8×8 patches. The BPFA is the best out of all configurations as expected. Using the learned bases by SPGL1 and by the RVM, we can see a great improvement as opposed to using the DCT illustrating the importance of learning a dictionary of bases. Other dictionaries are also used such as an inferred dictionary from all 10000 sections without obtaining significant improvements. Learning a dictionary of bases per section is more effective since it tailors the bases for that particular signal. Furthermore, a universal dictionary learned using many different signals might not be useful since the seismic wave field varies at different subsurface locations and different bases might be necessary each time.

Learning a dictionary of bases for each section helps reconstruction accuracy but we also wanted to investigate its computational time. We recorded the time it takes for the BPFA on 8×8 patches to run on the same one hundred and fifty sections that were

used for the RVM with DCT, the SPGL1 with DCT and POCS on 128×128 and 8×8 patches. We also recorded the computational time for the RVM with the BPFA bases on 8×8 and the SPGL1 with BPFA bases on 8×8 patches. The BPFA is faster than the RVM with DCT on 128×128 patches but slower than all other configurations. The SPGL1 with BPFA bases and the RVM with BPFA bases on 8×8 are faster than the BPFA and produce similar reconstruction accuracy. Nevertheless, they require that first the BPFA learns the bases.

Improving the computational time of the BPFA is essential and we thus proposed a Gibbs analysis for faster BPFA inference. We investigated how the initialisation of the dictionary of bases changes the reconstruction accuracy. Out of six initialisations, the initialisation with the Singular Value Decomposition (SVD) of the available data provides the best reconstruction accuracy. We also tracked how the reconstruction accuracy changes over time with more Gibbs rounds and iterations. We found that not all iterations are necessary, allowing us to halve the iterations in the last round without significant reduction in accuracy but with more than three hundred hours of speed up.

Due to the fact that the BPFA learns a data-driven model, it is possible to use it for other purposes. We wanted to investigate its behaviour on denoising seismic signals. Thus, we added Gaussian noise of varying levels on seismic signals and used the BPFA to remove it. We used the K-SVD for comparisons and illustrated that the BPFA provides higher levels of reconstruction accuracy in the expense of higher computational time. The higher reconstruction accuracy comes from the fact that the BPFA learns bases that capture higher frequencies as opposed to the K-SVD which learns more bases with lower frequency characteristics. We also performed simultaneous interpolation and denoising with success.

Operating in two dimensions is useful, however seismic signals are often represented in three dimensions due to the fact that receivers capture reflections over time. In three dimensions, more Gibbs iterations are necessary as illustrated with a three dimensional BPFA reconstruction which is partially completed. Thus, to maintain high reconstruction accuracy and practical computational time, we process three dimensional seismic signals in two dimension over time resulting in a pseudo 3D interpolation. Each two dimensional reconstruction can be processed independently and in parallel. Thus, the computational time is not increased if there is the infrastructure of computer hardware available.

By working in two dimensions, reconstruction of time slices is possible. Then, by sorting the time slices in the x-t domain, we can get a better understanding of the reconstruction accuracy of each algorithm. This is because we compute the Frequency Wavenumber (FK) domain of each reconstruction and visualise if there is any aliasing or

incoherent noise. As discussed, aliasing is avoided due to irregular sampling and thus only incoherent noise needs to be removed.

We experimented with two different types of receiver lines, those that pass close to the source and those that are far from it. From the experiments of receiver lines far from the source, the RVM with DCT bases on 128×128 patches obtained the best reconstruction accuracy for all percentages of receivers used. The BPFA on 8×8 patches obtained the best reconstruction accuracy overall for algorithms that operate on 8×8 patches and with fixed bases. Using the RVM with learned BPFA bases on 8×8 patches improved its performance, obtaining the second best performance overall.

For the closer to source receiver lines, the reconstruction is more challenging due to the steeper dips at the top centre of the signals. All algorithms that operated on 128×128 patches had issues reconstructing the top centre region of the receiver lines due to its very high variance. The basis functions used on 128×128 patches did not contain such localised components. On the other hand, the algorithms that operated on 8×8 patches reconstructed the region without problems due to the use of smaller basis functions that can capture finer, localised details. The algorithm with the best reconstruction accuracy in that region was the RVM with the DCT on 8×8 patches. The BPFA was not able to learn useful bases and thus the learned bases did not improve the performance of algorithms. Nevertheless, at the rest of the regions of the signal, the RVM with DCT on 128×128 patches obtained the best accuracy in general. The BPFA on 8×8 patches and the SPGL1 with DCT bases on 128×128 patches obtained high reconstruction accuracy as well. Note that the FK spectrum of the best performing algorithms did not exhibit any incoherent noise. Thus, if reconstruction accuracy is priority, the RVM with DCT should be used with different patch sizes depending on the region and the variance of the available data.

To get a better understanding of how algorithms operate, we analysed the relationship of the reconstruction accuracy with the variance of the available data. We identified and illustrated high variance regions in the receiver lines both far and close to the source with the latter having higher variance values in general. We used the Spearman's correlation coefficient to identify how the accuracy varies with the variance. In general, for all algorithms, the two are positively correlated, as the variance of the available data increases so does the reconstruction accuracy. However, when the variance is too large, the reconstruction accuracy suddenly decreases as discussed in the top centre of the close to source signals. The most positively correlated algorithm is also the best performing, that is the RVM with DCT on 128×128 patches. For all algorithms, when the variance of the available data is low, the reconstruction accuracy is bad and as the variance

increases it improves. Care is needed when splitting seismic signals into sections so as to include regions with enough variance but not too much to compromise the reconstruction accuracy.

With different levels of variance and with different configurations of algorithms, the accuracy of prediction for each receiver varies. In order to have the complete picture of reconstruction accuracy and risk associated with it, each prediction should be accompanied with a degree of certainty. Ideally, an uncertainty map should be produced showing the algorithm's confidence per prediction. This map should correlate well with the reconstruction error when we evaluate them on a known signal. The RVM and the BPFA are probabilistic data-driven models and can be used for creating uncertainty maps as opposed to algorithms that are ad hoc and just fill in gaps.

We have seen that the RVM's predictive variance does not perform well with basis functions that cover small regions in the input space such as the Haar wavelet transform. This leads to the degenerate case of a predictive variance equal to zero (if we ignore the noise variance). We used this to create a deep network of RVMs. However, the RVM with DCT was found to obtain better reconstruction accuracy and in turn better predictive variance (avoiding the degenerate case). We also used two other modifications of the RVM. The first is via augmentation where a new basis function is added at a missing receiver. The other depends on the change in the likelihood of the model which also depends on the variance of the neighbourhood of the predictions. We found that the latter two required a lot of parameter tuning and for fair comparisons we did not continue in their experimentation.

Furthermore, we used the BPFA to obtain reconstructions and at the same time to create uncertainty maps. This was achieved by exploiting its probabilistic nature. This is because, the inference stage involves random draws from the variables' distributions. By drawing different values and consequently predicting different receivers' values, it is possible to obtain a collection of predictions and then estimate a mean and a variance for each receiver. When the variance is small, BPFA trusts the value better and vice versa.

A comparison of the RVM's predictive variance and the BPFA's variance was undertaken on thousands of sections of time slices. The comparison was done using the Spearman's correlation coefficient which evaluates how correlated the uncertainty map is to the respective reconstruction error. We showed that BPFA produces better correlated uncertainty maps than the RVM for both far and close to the source receiver lines. Ranked scatter plots also illustrated this behaviour. Visualisations of various sections of time slices were also provided along with illustrations on a field data set.

For different sections, the Spearman's correlation coefficient varied. We thus wanted to investigate how it is affected by the variance of the available data as it was done for the reconstruction accuracy. For the BPFA, we have shown that in general, the correlation of the uncertainty maps does not heavily depend on the variance of the available data. On the other hand, the correlation of the RVM's uncertainty maps increases as the variance of the available data increases. Thus, care is again necessary when splitting the sections of time slices.

Since the Spearman's correlation coefficient for uncertainty maps varies, we need to take into consideration all uncertainty maps produced per time sample for a particular receiver. In our experiments, each receiver is composed of 500 time samples and we therefore needed to consider the uncertainty of a receiver's prediction in each of them. One option was to re-sort all the uncertainty maps in the x-t domain and visual the uncertainty as it was done for the predictions. However, this does not provide a quantitative measure.

Therefore, we decided to stack the uncertainty maps together over time and obtain an average uncertainty for each receiver from 500 uncertainty maps. Using this, we are able to get uncertainty maps that take into account all time samples per receiver. This was undertaken for both the BPFA and the RVM. In both cases, the correlation of the stacked uncertainty maps with the stacked reconstruction errors improved due to the amplification of the correctly correlated uncertainty regions and the removal of random uncertainties. Comparisons on numerous stacked sections illustrated that the BPFA is still better at creating uncertainty maps, albeit the RVM is also useful when stacking is undertaken.

8.1 Future work

Moving forward, there is further work to be done for better utilisation of Bayesian statistics and machine learning in seismic acquisition. Both the reconstruction accuracy and the computational time can be improved with appropriate future research. Faster software implementations of the RVM and the BPFA can help reduce the running time per iteration. This would allow the utilisation of more iterations in three dimensions, making it practical to work directly with three dimensional signals. In addition, this can also make the learning of bases on larger patch sizes easier.

At the moment, the BPFA learned a different dictionary of bases per percentage with varying accuracy. It would be interesting to investigate which percentage of receivers allows the BPFA to learn the most informative dictionary of bases. This can be done by learning bases for all percentages of receivers and then using them with the RVM and

the SPGL1 for a fixed percentage comparing the reconstruction accuracy on sections with different variance.

In addition, the variance analysis provided in this thesis was useful to identify regions of high and low variance and for locations where a configuration is more suitable than another. Further investigation can be performed by using clustering methods. Regions of seismic data can be clustered into various groups and then each algorithm evaluated in each group for its accuracy and uncertainty quantification. With this, future seismic data can be clustered and recommendation of which algorithm is more suitable for each cluster can be made improving the reconstruction accuracy and uncertainty quantification.

8.2 Conclusion

In conclusion, we proposed, improved and modified the RVM and the BPFA models in order to solve various tasks in seismic acquisition such as Compressive Sensing, denoising, feature learning and uncertainty quantification. The importance of constructing data-driven models for prediction and uncertainty quantification is growing. We have illustrated that Bayesian statistics and machine learning can be used in seismic data acquisition and imaging to provide accurate seismic signal reconstructions from under-sampled data. At the same time, they can provide uncertainty maps that could guide seismic survey design in the future. The importance of learning bases from seismic data is also growing. BPFA is an excellent example of how the reconstruction accuracy can be improved greatly with learned bases rather than by using pre-defined dictionaries. The learned bases and in general, the probabilistic data-driven models could be extended to other areas of seismic data acquisition, compression, classification, automated decision making and eventually to create new applications for the characterisation of seismic signals.

References

Abma, R., Howe, D., Foster, M., Ahmed, I., Tanis, M., Zhang, Q., Arogunmati, A., and Alexander, G. (2015). Independent simultaneous source acquisition and processing. *GEOPHYSICS*, 80(6):WD37–WD44.

Abma, R. and Kabir, N. (2006). 3D interpolation of irregular data with a POCS algorithm. *GEOPHYSICS*, 71(6):E91–E97.

Aharon, M., Elad, M., and Bruckstein, A. (2006). K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *IEEE Transactions of Signal Processing*, 54(11):4311–4322.

Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. (2003). An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43.

Baraniuk, R. G. and Steeghs, P. (2017). Compressive sensing: A new approach to seismic data acquisition. *The Leading Edge*, 36(8):642–645.

Beck, A. and Teboulle, M. (2009). A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal Imaging Sciences*, 2(1):183–202.

Beckouche, S. and Ma, J. (2014). Simultaneous dictionary learning and denoising for seismic data. *GEOPHYSICS*, 79(3):A27–A31.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Birgin, E. G., Mario, J., and Raydan, M. M. (2003). Inexact spectral projected gradient methods on convex sets. *IMA Journal on Numerical Analysis*, 23:539–559.

Blumensath, T. and Davies, M. (2008). Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, pages 629–654.

Bryan, K. and Leise, T. (2013). Making do with less: An introduction to compressed sensing. *SIAM Review*, 55(3):547–566.

Candes, E. and Tao, T. (2006). Near-Optimal Signal Recovery From Random Projections: Universal Encoding Strategies. *IEEE Transactions on Information Theory*, 52(12):5406–5425.

Candes, E. J. and Wakin, M. B. (2008). An Introduction to Compressive Sampling. *IEEE Signal Processing Magazine*, 25(2):21–30.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM Rev.*, 43(1):129–159.

Chen, Y., Ma, J., and Fomel, S. (2016). Double-sparsity dictionary for seismic noise attenuation. *GEOPHYSICS*, 81(2):V103–V116.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Dang, H.-P. (2016). *Bayesian nonparametrics approaches and dictionary learning for inverse problems in image processing*. Theses, Ecole Centrale de Lille.

Daubechies, I., Defrise, M., and De Mol, C. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math.*, 57(11):1413–1457.

Donoho, D. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306.

Duijndam, A. J. W. (1988a). Bayesian estimation in seismic inversion. part i: Principles. *Geophysical Prospecting*, 36(8):878–898.

Duijndam, A. J. W. (1988b). Bayesian estimation in seismic inversion. part ii: Uncertainty analysis. *Geophysical Prospecting*, 36(8):899–918.

Elad, M. and Aharon, M. (2006). Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745.

Faul, A. and Pilikos, G. (2016). The model is simple, until proven otherwise: How to cope in an ever-changing world. In *Data for Policy, Frontiers of Data Science for Government, 2016*.

Faul, A. C. and Tipping, M. E. (2001). Analysis of sparse bayesian learning. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 383–389, Cambridge, MA, USA. MIT Press.

Fjeldstad, T. and Grana, D. (2018). Joint probabilistic petrophysics-seismic inversion based on Gaussian mixture and Markov chain prior models. *GEOPHYSICS*, 83(1):R31–R42.

Fomel, S. and Liu, Y. (2010). Seislet transform and seislet frame. *GEOPHYSICS*, 75(3):V25–V38.

Foucart, S. and Rauhut, H. (2013). *A Mathematical Introduction to Compressive Sensing*.

Gao, J., Sacchi, M. D., and Chen, X. (2013). A fast reduced-rank interpolation method for prestack seismic volumes that depend on four spatial dimensions. *GEOPHYSICS*, 78(1):V21–V30.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6):721–741.

Ghahramani, Z. and Hinton, G. E. (1997). The EM algorithm for mixtures of factor analyzers. Technical report.

Griffiths, T. L. and Ghahramani, Z. (2011). The Indian Buffet Process: An introduction and review. *J. Mach. Learn. Res.*, 12:1185–1224.

Gülünay, N. (2003). Seismic trace interpolation in the Fourier transform domain. *GEOPHYSICS*, 68(1):355–369.

Hennenfent, G. and Herrmann, F. J. (2008). Simply denoise: Wavefield reconstruction via jittered undersampling. *GEOPHYSICS*, 73(3):V19–V28.

Herrmann, F. J. and Hennenfent, G. (2008). Non-parametric seismic data recovery with curvelet frames. *Geophysical Journal International*, 173(1):233–248.

Hinton, G. E. (2002). Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800.

Ji, S., Xue, Y., and Carin, L. (2008). Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56(6):2346–2356.

Jia, Y. and Ma, J. (2017). What can machine learning do for seismic data processing? An interpolation application. *GEOPHYSICS*, 82(3):V163–V177.

Jia, Y., Yu, S., and Ma, J. (2018). Intelligent interpolation by Monte Carlo machine learning. *GEOPHYSICS*, 83(2):V83–V97.

Jiang, T., Gong, B., Qiao, F., Jiang, Y., Chen, A., Hren, D., and Meng, Z. (2017). Compressive seismic reconstruction with extended POCS for arbitrary irregular acquisition. *SEG Technical Program Expanded Abstracts 2017*, pages 4272–4277.

Jingjie, C., Yanfei, W., and Benfeng, W. (2015). Accelerating seismic interpolation with a gradient projection method based on tight frame property of curvelet. *Exploration Geophysics*, 46(3):253–260.

Kabir, M. N. and Verschuur, D. (1995). Restoration of missing offsets by parabolic Radon transform. *Geophysical Prospecting*, 43(3):347–368.

Kazemi, N., Bongajum, E., and Sacchi, M. D. (2016). Surface-consistent sparse multichannel blind deconvolution of seismic signals. *IEEE Transactions on Geoscience and Remote Sensing*, 54(6):3200–3207.

Kong, D. and Peng, Z. (2015). Seismic random noise attenuation using shearlet and total generalized variation. *Journal of Geophysics and Engineering*, 12(6):1024.

Kreimer, N. and Sacchi, M. D. (2011). A tensor higher order singular value decomposition (HOSVD) for prestack simultaneous noise reduction and interpolation. *SEG Technical Program Expanded Abstracts 2011, 3069-3074.*

References

Kreimer, N. and Sacchi, M. D. (2012). A tensor higher order singular value decomposition for prestack seismic data noise reduction and interpolation. *GEOPHYSICS*, 77(3):V113–V122.

Kumar, R., Silva, C. D., Akalin, O., Aravkin, A. Y., Mansour, H., Recht, B., and Herrmann, F. J. (2015). Efficient matrix completion for seismic data reconstruction. *GEOPHYSICS*, 80(5):V97–V114.

Kutscha, H. and Verschuur, D. (2016). The utilization of the double focal transformation for sparse data representation and data reconstruction. *Geophysical Prospecting*, 64(6):1498–1515.

Liang, J., Ma, J., and Zhang, X. (2014). Seismic data restoration via data-driven tight frame. *GEOPHYSICS*, 79(3):V65–V74.

Liu, B. and Sacchi, M. D. (2004). Minimum weighted norm interpolation of seismic records. *GEOPHYSICS*, 69(6):1560–1568.

Liu, Y. and Fomel, S. (2010). Oc-seislet: Seislet transform construction with differential offset continuation. *GEOPHYSICS*, 75(6):WB235–WB245.

Malinverno, A. and Briggs, V. A. (2004). Expanded uncertainty quantification in inverse problems: Hierarchical bayes and empirical bayes. *GEOPHYSICS*, 69(4):1005–1016.

Mallat, S. and Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415.

Morrison, J. L. P. (2013). Signals and systems at Montana Tech. *Electrical Engineering Books. 2.*

Mosher, C. C., Keskula, E., Kaplan, S. T., Keys, R. G., Li, C., Ata, E. Z., Morley, L. C., Brewer, J. D., Janiszewski, F. D., Eick, P. M., Olson, R. A., and Sood, S. (2012). Compressive Seismic Imaging, *SEG Technical Program Expanded Abstracts 2012*, pages 1–5.

Naghizadeh, M. and Sacchi, M. D. (2007). Multistep autoregressive reconstruction of seismic records. *GEOPHYSICS*, 72(6):V111–V118.

Naghizadeh, M. and Sacchi, M. D. (2010). Beyond alias hierarchical scale curvelet interpolation of regularly and irregularly sampled seismic data. *GEOPHYSICS*, 75(6):WB189–WB202.

Natarajan, B. K. (1995). Sparse Approximate Solutions to Linear Systems. *SIAM Journal on Computing*, 24(2):227–234.

Oropeza, V. and Sacchi, M. (2011). Simultaneous seismic data denoising and reconstruction via multichannel singular spectrum analysis. *GEOPHYSICS*, 76(3):V25–V32.

Paisley, J. and Carin, L. (2009). Nonparametric factor analysis with beta process priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 777–784, New York, NY, USA. ACM.

Pérez, D. O., Velis, D. R., and Sacchi, M. D. (2013). High-resolution prestack seismic inversion using a hybrid FISTA least-squares strategy. *GEOPHYSICS*, 78(5):R185–R195.

Pilikos, G. (2014). Signal reconstruction using compressive sensing. *MPhil Scientific Computing Thesis, University of Cambridge*.

Porsani, M. J. (1999). Seismic trace interpolation using halfstep prediction filters. *GEOPHYSICS*, 64(5):1461–1467.

Rasmussen, C. E. and Quiñonero Candela, J. (2005). Healing the Relevance Vector Machine through augmentation. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 689–696.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the Twenty-eight International Conference on Machine Learning*.

Ronen, J. (1987). Wave equation trace interpolation. *GEOPHYSICS*, 52(7):973–984.

Sacchi, M., Ulrych, T., and Walker, C. (1998). Interpolation and extrapolation using a high resolution discrete Fourier transform. *IEEE Transactions on Signal Processing*, 46(1):31–38.

SEG (2018a). http://seg.org/news-resources/research-and-data/seam, Last accessed 8 January 2018.

SEG (2018b). http://wiki.seg.org/wiki/parihaka-3d, Last accessed 2 February 2018.

Shahidi, R., Tang, G., Ma, J., and Herrmann, F. J. (2013). Application of randomized sampling schemes to curvelet based sparsity promoting seismic data recovery. *Geophysical Prospecting*, 61(5):973–997.

Sheriff, R. and Geldart, L. (1982). *Exploration Seismology*. Cambridge University Press.

Siahsar, M. A. N., Gholtashi, S., Kahoo, A. R., Chen, W., and Chen, Y. (2017). Data driven multitask sparse dictionary learning for noise attenuation of 3D seismic data. *GEOPHYSICS*, 82(6):V385–V396.

Spitz, S. (1991). Seismic trace interpolation in the f-x domain. *GEOPHYSICS*, 56(6):785–794.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Stanton, A., Kreimer, N., Bonar, D., Naghizadeh, M., and Sacchi, M. (2012). A comparison of 5D reconstruction methods. *SEG Technical Program Expanded Abstracts 2012, 1-5*.

Stanton, A., Sacchi, M. D., Abma, R., and Stein, J. A. (2015). Mitigating artifacts in Projection Onto Convex Sets interpolation. *SEG Technical Program Expanded Abstracts 2015, 3779-3783*.

Sutton, R. S. and Barto, A. G. (1998). *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition.

Tian, X., Zhang, K., and Li, Z. (2017). Seismic data denoising based on online dictionary learning algorithm, *SEG Technical Program Expanded Abstracts 2017*, pages 5105–5109.

Tibshirani, R. (1994). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.

Tipping, M. E. (2001). Sparse Bayesian learning and the Relevance Vector Machine. *J. Mach. Learn. Res.*, 1:211–244.

Tipping, M. E. and Faul, A. (2003). Fast marginal likelihood maximisation for sparse Bayesian models. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, pages 3–6.

Trad, D. O., Ulrych, T. J., and Sacchi, M. D. (2002). Accurate interpolation with high resolution time variant Radon transforms. *GEOPHYSICS*, 67(2):644–656.

Trickett, S., Burroughs, L., Milton, A., Walton, L., and Dack, R. (2010). Rank reduction based trace interpolation, *SEG Technical Program Expanded Abstracts 2010*, pages 3829–3833.

Tropp, J. and Gilbert, A. (2007). Signal recovery from random measurements via Orthogonal Matching Pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666.

Turquais, P., Asgedom, E., and Soellner, W. (2017). Structured dictionary learning for interpolation of aliased seismic data. *SEG Technical Program Expanded Abstracts 2017*, pages 4257–4261.

Turquais, P., Asgedom, E. G., Sollner, W., and Otnes, E. (2015). Dictionary learning for signal-to-noise ratio enhancement. *SEG Technical Program Expanded Abstract 2015, 4698-4702*.

Ulrych, T. J., Sacchi, M. D., and Woodbury, A. (2001). A Bayes tour of inversion: A tutorial. *GEOPHYSICS*, 66(1):55–69.

van den Berg, E. and Friedlander, M. P. (2009). Probing the Pareto Frontier for Basis Pursuit Solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA. ACM.

Wang, D., Saab, R., Yilmaz, O., and Herrmann, F. J. (2008). Bayesian wavefield separation by transform domain sparsity promotion. *GEOPHYSICS*, 73(5):A33.

Xu, S., Zhang, Y., Pham, D., and Lambaré, G. (2005). Antileakage Fourier transform for seismic data regularization. *GEOPHYSICS*, 70(4):V87–V95.

Yu, S., Ma, J., and Osher, S. (2016). Monte Carlo data-driven tight frame for seismic data recovery. *GEOPHYSICS*, 81(4):V327–V340.

Yu, S., Ma, J., Zhang, X., and Sacchi, M. D. (2015). Interpolation and denoising of high-dimensional seismic data by learning a tight frame. *GEOPHYSICS*, 80(5):V119–V132.

Zhou, M., Chen, H., Paisley, J., Ren, L., Li, L., Xing, Z., Dunson, D., Sapiro, G., and Carin, L. (2012). Nonparametric Bayesian dictionary learning for analysis of noisy and incomplete images. *IEEE Transactions on Image Processing*, 21(1):130–144.

Zhou, M., Chen, H., Ren, L., Sapiro, G., Carin, L., and Paisley, J. W. (2009). Non-parametric Bayesian dictionary learning for sparse image representations. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22*, pages 2295–2303.

Zhu, H., Li, S., Fomel, S., Stadler, G., and Ghattas, O. (2016). A Bayesian approach to estimate uncertainty for full-waveform inversion using a priori information from depth migration. *GEOPHYSICS*, 81(5):R307–R323.

Zhu, L., Liu, E., and McClellan, J. H. (2015). Seismic data denoising through multiscale and sparsity-promoting dictionary learning. *GEOPHYSICS*, 80(6):WD45–WD57.

Zhu, L., Liu, E., and McClellan, J. H. (2017). Sparse-promoting full-waveform inversion based on online orthonormal dictionary learning. *GEOPHYSICS*, 82(2):R87–R107.

Zwartjes, P. M. and Sacchi, M. D. (2007). Fourier reconstruction of nonuniformly sampled, aliased seismic data. *GEOPHYSICS*, 72(1):V21–V32.